UD Japanese に基づく国語研長単位解析系の構築

松田 寛 †
† 株式会社リクルート Megagon Labs, Tokyo, Japan
hiroshi_matsuda@megagon.ai

大村 舞 ‡　浅原 正幸 ‡
‡ 国立国語研究所
{mai-om, masayu-a}@ninjal.ac.jp

概要

国語研の規程では，長単位は文節内部を自立語部分と付属語部分に分割する形で定義されるが，固有表現や複合辞・連語については個別の規則が適用される．また，長単位品詞は，短単位品詞の「名詞-普通名詞-副詞可能」「動詞-非自立可能」等の用法の曖昧性を，実際の文脈における用法で解決する必要がある．本研究では，Universal Dependencies に基づく依存構造解析モデルを拡張し，形態素解析器の短単位出力を長単位化する手法を評価した．同時に，用法に基づく 17 種の UPOS 推定結果，固有表現抽出結果，長単位末尾の形態素情報を組み合わせた長単位品詞判定規則を構築し，従来手法を上回る 97.2 ポイントの長単位品詞推定精度を得た．

1　はじめに

日本語の自然言語処理は前段の形態素解析器で入力テキストを単語分割することが多い[1)]．形態素解析器は単語分割と同時に，品詞推定，辞書形・正規形への正規化，読み付与等を行い，応用側は形態素解析器から単語と属性情報の列を容易に取得できる．ただし形態素解析器は一般に処理速度を優先する設計をとるため，文中での実際の単語の使用における意味や統語的役割の曖昧性解消といった計算負荷の高い処理は行われない．また複合語は組み合わせが膨大なため辞書に登録する範囲は制限される．

一方，日本語の文節は文の統語構造や単語の意味を解釈するための重要な手がかりであり，形態素解析器では解決されない品詞曖昧性解消や複合辞認定といった高度な処理では，一般に文節の考慮が必要とされる．国語研長単位 [2] は，文節から自立語部分と付属語部分を規則的に分割したものに，さらに用法に基づく品詞を付与し，複合語を単位とする正規化等を施したものである．本研究では，形態素解析器と同様の手軽さで利用可能な，国語研長単位に基づいた解析系の構築を目指す．

1） 深層学習の素性化には Sentencepiece[1] など形態素解析とは異なる単語分割手法も用いられている．

2　長単位境界と品詞認定

日本語 NLP の単語単位は形態素解析器に同梱されている辞書に基づき解析されており，JUMAN 辞書 [3] や UniDic 辞書 [4] に基づくものなどが広く用いられている．後者の UniDic 辞書は，国語研内で規定されている国語研短単位 [5] に基づく．短単位は，字種ごとに定義されている国語研最小単位に基づき，同字種最小単位の 1 回結合までを形態論に基づく語の単位として定義する．国語研では，日本語の一般的な統語分析の単位として用いられる文節境界に基づく単位として，国語研長単位 [2] も定義している．長単位の認定は，文節の認定を行った上で，各文節の内部を規則に従って自立語部分と付属語部分に分割していくという手順で行う．長単位では，複合的な機能表現を含めて，複合語を構成要素に分割することなく全体で一つとして扱う．短単位と長単位では，品詞認定手法についても異なり，短単位の品詞認定においては，「可能性に基づく品詞体系」として，形態に基づいて可能な品詞を枚挙（例：「名詞-普通名詞-サ変形状詞可能」など）するために統語的な用法の本質的な解決は行わない．長単位の品詞認定においては，「用法に基づく品詞体系」として，複合化した結果の単位が文脈内でどのように振る舞うかに基づき，品詞を認定する．

また本研究で用いる Universal Dependencies(UD) [6] はアノテーションする単位として「**Syntactic Word** を用いること」 [6] と定義している．UD Japanese r2.6-2.8 [7] においては，短単位に基づく treebank を構成していたが，短単位は Syntactic Word としてはふさわしくない点が指摘されている [8]．UD Japanese r2.9 においては，新たに長単位に基づく treebank を構成した [9]．長単位は前述のとおり Syntactic Word に近いと推定されるが，工学的に有用であるかどうかの検証まではされていない．

長単位解析系として，Comainu [10] が公開されている．Comainu 内部では，MeCab [11] の解析結果の短単位形態素情報を素性化し，さらに CRF で系列ラベリングを行うことで長単位解析を実現している．

spaCy は Universal Dependencies に基づく自然言語処理フレームワークであり，Non-Monotonic Arc-Eager Transition System[12] [2] ベースの parser と transformers モデルの間で Gradient を共有して学習を行うことで高い解析精度を実現している．松田らは，サブトークン結合用の依存関係ラベルを用いて形態素解析器が過剰分割したトークンを依存構造解析と同時にまとめ上げる手法を提案し，日本語 UD 解析モデルの精度を改善した [13]．後に，spaCy にもサブトークン結合専用ラベル **subtok** を用いたトークンまとめ上げ機能が組み込まれている．

本研究では，spaCy の各種解析機能を使用して，形態素解析器出力の長単位へのまとめ上げを含めた長単位解析モデルを構築した．

3 長単位認定モデルの構築

3.1 spaCy 日本語モデルの拡張

2021 年 11 月に公開された spaCy 日本語 transformers モデルは，transformers 層に BERT 事前学習済みモデル [14] を使用することで，依存構造解析・固有表現抽出・UD 品詞推定の各精度を大きく向上した．本研究では，spaCy で長単位解析系を構築するために，spaCy 日本語 transformers モデルに対して次の拡張を行った．図 1 に spaCy の Pipeline 構成を示す．

形態素解析器 複合辞出力モードを指定
学習データの加工 形態素解析器でテキストを再解析，長単位が分割される区間に **subtok** ラベルを適用
transformers モデル UniDic 短単位+wordpiece の BERT 事前学習済みモデルを指定
長単位解析コンポーネントの追加 merge_subtokens による長単位認定 (**subtok** 連続区間のまとめ上げ)，luw_xpos_tagger による長単位の正規化+品詞判定

3.2 長単位と固有表現の関係

松田らは，UD_Japanese-GSD r2.6 に対して，spaCy が採用する OntoNotes5 の体系に基づく正解固有表現ラベルを付与した [15]．本研究では，UD_Japanese-GSD の正解固有表現ラベルと GSDLUW のアライメント処理を行い，長単位ベースの固有表現抽出モデルの学習に使用した．固有表現抽出結果は長単位品詞判定ルールで参照される．長単位 (LUW) と固有表現 (NE) のスパンの対応関係の統計を表 1 に示す．全固有表現のうち 2/3 は単一長単位と厳密に対応づいているため，アライメント後の固有表現ラベルで学習した長単位固有表現抽出モデルはある程度機能すると期待する．

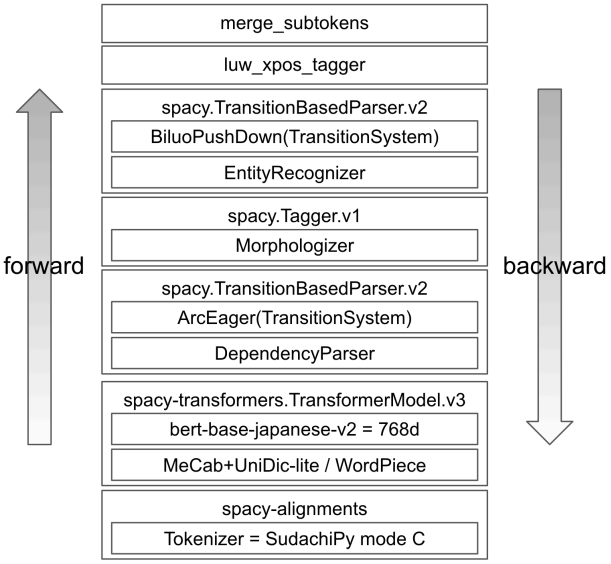

図 1 spaCy 長単位解析モデルの Pipeline 構成

表 1 長単位区間と固有表現区間の対応

UD_Japanese-GSD に含まれる NE 数		12,327
単一 LUW が NE を内包		10,785
非 NE 要素を包まない	8,345	
前方のみに非 NE 要素を包む	267	
後方のみに非 NE 要素を包む	2,088	
前後両方に非 NE 要素を包む	85	
複数 LUW が NE を内包		1,141
非 NE 要素を包まない	844	
前方のみに非 NE 要素を包む	296	
後方のみに非 NE 要素を包む	1	
前後両方に非 NE 要素を包む	0	
単一 LUW が複数 NE に対応		401

固有表現が単一長単位に対応しない文脈の例を表 2 に示す．長単位が文節構造から統語的規則で認定されるスパンであるのに対して，固有表現は所与のカテゴリ定義に該当する最短のスパンが認定されている．今後の課題として，この両者の比較から，固

2) [12] の Table 1 最下部の Left-arc 更新後の状態の記述は誤り．正しくは $(\sigma, b|\beta, \mathbf{A}(s) = b, \mathbf{S})$ となる．また **S** は **Unshift** で $\mathbf{S}(s) = 1$ に，**Left-arc** で $\mathbf{S}(b) = 0$ にリセットされる．spaCy の内部実装では **Reduce** と **Unshift** は統合 (head 有無で区別) され，さらに文区切りアクション **Break** が追加されている．

有表現が文中で様々な接辞類をまとって長単位化する際に，どのような意味の具象化・抽象化が行われるかを分析し，各固有表現カテゴリの典型的な修飾要素の類型化につなげていきたい．

表 2 固有表現が単一長単位に対応しない例

長単位区間	固有表現区間
区切り基準の齟齬	
8 月 / 11 日生まれ	8 月 11 日
当駅-下関駅間	下関駅
広島県 / 広島市/ 中区 / 中島町	広島県広島市中区中島町
形状詞化・副詞化	
ツイッター特有 (形状詞)	ツイッター
東西冷戦中 (副詞)	東西冷戦
具象化・抽象化・総称化	
3 年 3 カ月ぶり	3 年 3 カ月
約二百人	二百人
北西大西洋	大西洋
祐介たち	祐介
人と役職・敬称・経歴	
斉藤惇社長	斉藤惇 / 社長
ダース・ベイダー役	ダース・ベイダー
矢部さん	矢部
元広島県議会議員	広島県議会 / 議員

4 長単位正規化規則の構築

形態素解析処理で使用する辞書の多くは，出現形に対応する正規形や辞書形が登録されており，短単位出力であればそれらを使用して正規化を行うことができる．一方，長単位の場合は，用言系の複合辞は最終形態素のみを正規化する等の特別な処理が必要となる．また国語研長単位の正規化の規程は場合分けが複雑なため，使用する形態素解析辞書に応じた正規形変換処理が必要となる．本研究では Sudachi 辞書 [16] 向けの変換規則を，次の 3 レベルの優先度に分けて構築して使用した．

1. CharFallback (上位規則未適用時) 数字とアルファベットの全半角統制・助数詞の正規化
2. Any (上位規則未適用時) {為る: する, こと: 事, つき: 付き, いずれ: 何れ, ただし: 但し}
3. Lexical
3-1. Mono (LUW=SUW) {有る: ある, 居る: いる, 得る: える, 成る: なる, 見る: みる, 出来る: できる, です: だ, 良い: よい, 無い: ない, ここ: 此処, そこ: 其処, どこ: 何処, あそこ: 彼処, あれ: 彼れ, これ: 此れ, それ: 其れ, どれ: 何れ, ため: 為, とも: 共, また: 又}
3-2. Init (最終 SUW 以外) 活用語以外を正規化
3-3. Last (最終 SUW) {よる: 因る, おる: 居る}

5 長単位品詞判定規則の構築

日本語の活用型は出現頻度が低いものがあり，全ての活用型の品詞タグを備えた treebank を構築することは一般に難しい．UD_Japanese-GSDLUW および UD_Japanese-BCCWJLUW には小椋らの規程集 [2] で定義される活用型のうち「動詞-一般-上一段-ハ行」「助動詞-助動詞-ドス」など 27 種の活用型が含まれていないため，機械学習ベースで長単位品詞推定を行うには，学習データに出現しない未知の活用型について配慮が必要となる．本研究では，機械学習モデルの利用を，短単位から長単位へのまとめ上げ，および，Universal Dependencies の用法に基づく 17 種類の UPOS 推定のみに使用し，長単位品詞 (LUW_XPOS) については簡易な判定規則を構築した．(判定規則の具体例を Appendix A に示す)

判定規則は次の 5 種類の情報を参照可能とした．

LUW_UPOS 長単位に対する UPOS 推定結果
SUW_XPOS 長単位末尾の形態素の短単位品詞
SUW_LEMMA 長単位末尾の形態素の lemma
NE 固有表現抽出結果のカテゴリ
Priority previous SUW_XPOS 長単位末尾形態素よりも優先して考慮する末尾直前の形態素の短単位品詞

長単位品詞判定規則は，GSDLUW・BCCWJLUW とそれらの短単位版との共起統計から人手で抽出し，優先度を次の 5 レベルに分けて実装した．

1. Fallback 最も優先度が低いフォールバックルール．LUW_UPOS を参照し，個々の LUW_UPOS に対応する典型的な長単位品詞が存在する場合はそれを適用する．また，可能性に基づく品詞を既定の用法の長単位品詞に変換する規則も適用する．いずれの規則にも該当しない場合は，短単位品詞をそのまま長単位品詞として用いる．
2. NE NE と LUW_UPOS に加えて，PERSON カテゴリでは Priority previous SUW_XPOS まで参照して，名詞-固有名詞の細分類を判定する．
3. Single 長単位が単一 SUW で構成される場合に適用される．SUW_XPOS と LUW_UPOS の組み合わせのみを参照する．
4. Multi 長単位が複数の SUW で構成される場合に適

用される．SUW_XPOS・LUW_UPOS に加え，特に接辞系の品詞の用法判定のために SUW_LEMMA を参照する．なお，**Single** と **Multi** をまとめたものを **Base** と呼ぶ．

5. Recovery 最も優先度が高いリカバリルール．SUW_XPOS・LUW_UPOS に加え，必要に応じて SUW_LEMMA まで参照することで，形態素解析器の典型的な誤解析から正しい長単位品詞を得る．

6 実験

Comainu と spaCy の長単位解析精度を比較する実験を行った．学習データには UD_Japanese-GSDLUW r2.9(修正版) に固有表現正解ラベルを追加したものを使用し，train セット 7,027 文をモデルの訓練に，dev セット 506 文をパラメータと判定規則のチューニングに，test セットを精度評価に用いた．Comainu のライブラリとモデルは v0.72 を用いて評価した．なお，Comainu のモデルを GSDLUW で学習・評価する追加実験では，公式モデルに比べ全般に若干精度が低下していたため記載を省略した．spaCy は設定ファイルで transformers モデルを指定し，cl-tohoku/bert-base-japanese-v2 および SudachiPy[17] モード C を指定した．spaCy のモデル学習には GSDLUW を SudachiPy で再解析したものを使用した．この再解析では，1 つの長単位が複数形態素に分割される場合，分割区間の主辞を区間末尾のサブトークンとし，各サブトークンは直後のサブトークンに依存する形として，長単位再構成用の **subtok** ラベルを付与した．これ以外の区切りの不整合は無視し，GSDLUW の区切りを優先して用いた．

長単位トークン認定，長単位正規化，長単位品詞推定の精度比較結果を表 3 に示す．

長単位トークン認定では，Comainu に対して spaCy では誤りが約半分まで低減された．これは，spaCy が transformers を介して文全体の構文構造を考慮したトークンまとめ上げを行っているのに対して，Comainu の CRF モデルは局所文脈しか考慮できないことが制約になっている可能性がある．

長単位正規化においては，spaCy の精度が Comainu を大きく上回った．これは正規系変換規則の効果に加えて，SudachiPy モード C が複合動詞の正規化に対応していることによる貢献が大きい．spaCy の誤解析の大半は英単語・数値・住所だったが，英単語・数値は全半角統制ポリシーが辞書登録語彙に依存していること，住所は SudachiPy モード C が住所全体を一形態素としていることが原因であった．SudachiPy のさらなる改良項目として，活用を維持したままでの正規化，住所と住所以外の解析モードを独立に指定可能にすること，が考えられる．

表 3 長単位解析精度の比較

	P	R	F1
Token:			
Comainu	97.6	96.9	97.3
spaCy$_{GSDLUW}$	98.7	98.5	98.6
Lemma:			
Comainu	88.9	88.2	88.6
spaCy$_{GSDLUW}$	96.0	95.8	95.9
LUW_XPOS:			
Comainu	96.2	95.4	95.8
spaCy$_{GSDLUW}$			
Rules$_{Fallback}$	91.0	90.8	90.9
+ Rules$_{Single}$	93.4	93.3	93.3
+ Rules$_{Base}$	96.5	96.4	96.4
+ Rules$_{Base+Recovery}$	96.7	96.6	96.7
+ Rules$_{Base+Recovery+NE}$	97.3	97.2	97.2

長単位品詞推定精度は，spaCy に基本判定規則を組み合わせることで Comainu を上回り，さらに誤解析回復規則と固有表現抽出結果を参照する規則を加えることで最良の結果が得られた．

長単位解析精度の差分については，正解トークンを用いた追加実験でも同様の傾向が確認された．

表 4 長単位モデルの依存構造解析精度

	UPOS	UAS	LAS
spaCy$_{GSDLUW}$	97.7	93.7	92.8

表 4 に長単位モデルの依存構造解析精度を示す．長単位品詞判定規則が機能する上で必要な水準の LUW_UPOS 推定精度が得られている．Unlabeled Attachment Score，および，Labeled Attachment Score は一般的な日本語モデルと同水準にあり，長単位へのまとめ上げに使用する **subtok** ラベルが他の依存関係ラベルと調和して機能していることが伺える．

7 まとめ

spaCy を用いた長単位解析系を実装し，長単位トークン認定・長単位品詞推定・長単位正規化において従来手法を大きく上回る精度を得た．依存構造解析は UAS・LAS ともに高水準であった．

謝辞

本研究は株式会社リクルート-国立国語研究所共同研究（研究課題名「日本語版 Universal Dependencies に基づく日本語依存構造解析モデルの研究開発」2019-2021 年度）によるものです．長単位解析系の実装にあたって，理化学研究所の松本裕治先生から多くの助言を頂きました．本研究で使用した spaCy モデルのベースとなった spaCy 日本語 transformers モデルは，spaCy 日本語コミュニティーの Gitter を通じて，参加者の皆様と基本設計を行いました．SudahiPy および Sudachi 辞書の利用においては，株式会社ワークスアプリケーションズ・エンタープライズ徳島人工知能 NLP 研究所の皆様に多大なご協力を頂きました．本研究の解析モデルは，explosion/spaCy・huggingface/transformers・cl-tohoku/bert-base-japanese-v2・MeCab など高性能なオープンプロダクトを利用することで効率的に実装することができました．本研究をご支援いただいた皆様に，心より感謝申し上げます．

参考文献

[1] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[2] 小椋秀樹, 小磯花絵, 冨士池優美, 宮内佐夜香, 小西光, 原裕. 『現代日本語書き言葉均衡コーパス』形態論情報規程集 第 4 版（上）. Technical report, 国立国語研究所内部報告書, 2011.

[3] Hajime Morita, Daisuke Kawahara, and Sadao Kurohashi. Morphological analysis for unsegmented languages using recurrent neural network language model. In **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**, pp. 2292–2297. Association for Computational Linguistics, September 2015.

[4] Yasuharu Den, Junpei Nakamura, Toshinobu Ogiso, and Hideki Ogura. A proper approach to Japanese morphological analysis: Dictionary, model, and evaluation. In **Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)**. European Language Resources Association (ELRA), May 2008.

[5] 小椋秀樹, 小磯花絵, 冨士池優美, 宮内佐夜香, 小西光, 原裕. 『現代日本語書き言葉均衡コーパス』形態論情報規程集 第 4 版（下）. Technical report, 国立国語研究所内部報告書, 2011.

[6] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal Dependencies v1: A multilingual treebank collection. In **Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)**. European Language Resources Association (ELRA), May 2016.

[7] Mai Omura and Masayuki Asahara. UD-Japanese BCCWJ: Universal Dependencies annotation for the Balanced Corpus of Contemporary Written Japanese. In **Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)**. Association for Computational Linguistics, November 2018.

[8] Yugo Murawaki. On the Definition of Japanese word. June 2019. arXiv: 1906.09719 [cs.CL].

[9] Mai Omura, Aya Wakasa, and Masayuki Asahara. Word Delimitation Issues in UD Japanese. In **Proceedings of the Fifth Workshop on Universal Dependencies (UDW, SyntaxFest 2021)**, 2021.

[10] 小澤俊介, 内元清貴, 伝康晴. 長単位解析器の異なる品詞体系への適用. 自然言語処理, Vol. 21, No. 2, 2014.

[11] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to Japanese morphological analysis. In **Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing**. Association for Computational Linguistics, July 2004.

[12] Matthew Honnibal and Mark Johnson. An improved non-monotonic transition system for dependency parsing. In **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[13] 松田寛, 大村舞, 浅原正幸. 短単位品詞の用法曖昧性解決と依存関係ラベリングの同時学習. 言語処理学会 第 25 回年次大会 発表論文集, 2019.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1**. Association for Computational Linguistics, June 2019.

[15] 松田寛, 若狭絢, 山下華代, 大村舞, 浅原正幸. Ud japanese gsd の再整備と固有表現情報付与. 言語処理学会 第 26 回年次大会 発表論文集, 2020.

[16] 坂本美保, 川原典子, 久本空海, 髙岡一馬, 内田佳孝. 形態素解析器『sudachi』のための大規模辞書開発. 言語資源活用ワークショップ 2018 発表論文集, 2018.

[17] Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. Sudachi: a Japanese tokenizer for business. In **Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)**. European Language Resources Association (ELRA), May 2018.

Appendix A　長単位品詞判定規則

長単位品詞判定規則は，Rules$_{_Recovery}$，Rules$_{_Base}$，Rules$_{_NE}$，Rules$_{_Fallback}$ の順に適用し，可能性に基づく品詞が残存する場合は既定の長単位品詞にフォールバックさせる．Rules$_{_Base}$ は，長単位が単一 SUW で構成される場合の Rules$_{_Single}$(33 エントリ) と，複数 SUW で構成される場合の Rules$_{_Multi}$(34 エントリ) に分けて管理され，Rules$_{_Recovery}$(33 エントリ) は形態素解析器の誤解析回復に使用される．

表 5　SUW 情報と LUW_UPOS を参照する長単位品詞判定規則の一部

SINGLE_SUW_XPOS	SINGLE_SUW_LEMMMA or LUW_UPOS : LUW_XPOS
助詞-係助詞	ADP: 助詞-係助詞, SCONJ: 助詞-接続助詞
助詞-副助詞	ADP: 助詞-副助詞, PART: 助詞-副助詞
助詞-準体助詞	ADP: 助詞-格助詞, PART: 助詞-終助詞, SCONJ: 助詞-準体助詞
名詞-固有名詞-一般	PROPN: 名詞-固有名詞-一般
形容詞-一般-形容詞	ADJ: 形容詞-一般-形容詞
感動詞-フィラー	INTJ: 感動詞-フィラー
接続詞	ADP:{て: 助詞-副助詞, *: 助詞-格助詞}, AUX: 助動詞-助動詞-ダ
LAST_SUW_XPOS	**LAST_SUW_LEMMMA or LUW_UPOS : LUW_XPOS**
助動詞-助動詞-タ	ADP: 助詞-格助詞, SCONJ: 助詞-接続助詞
動詞-一般-サ行変格	ADP:{かんする|たいする|対する|関する|際する: 助詞-格助詞}
動詞-一般-五段-ラ行	ADP: 助詞-格助詞, VERB: {よる: 助詞-格助詞}
動詞-非自立可能-上一段-ア行	SCONJ: 助動詞-上一段-ア行
接尾辞-動詞的-五段-カ行	VERB: 動詞-一般-五段-カ行

表 6　Rules$_{_NE}$: 固有表現カテゴリに基づく長単位品詞判定規則

NE Category	LUW_XPOS	Excluding last XPOS	Priority previous XPOS
GPE, LOC	名詞-固有名詞-地名-一般	名詞-普通名詞-一般	-
MONEY, PERCENT	名詞-数詞	名詞-普通名詞-一般	-
NORP, ORG	名詞-固有名詞-一般	名詞-普通名詞-一般	-
PERSON, TITLE_AFFIX	名詞-固有名詞-人名-一般	名詞-普通名詞-一般	名詞-固有名詞-人名-姓
PET_NAME	名詞-固有名詞-一般	-	-

表 7　Rules$_{_Fallback}$: LUW_UPOS に基づく長単位品詞判定規則

LUW_UPOS	LUW_XPOS	LUW_UPOS	LUW_XPOS	LUW_UPOS	LUW_XPOS
ADJ	形状詞-一般	DET	連体詞	NUM	名詞-数詞
ADV	副詞	INTJ	感動詞-一般	PRON	代名詞
CCONJ	接続詞	NOUN	名詞-普通名詞-一般	SYM	補助記号-一般

表 8　実験条件詳細

UD_Japanese-GSDLUW+NE r2.9	(train=7,027 dev=506 test=542 sentence)
UD_Japanese-BCCWJLUW r2.9	(train=40,801 dev=8,427 test=7,881, only used for rule construction)
Comainu 0.72	MeCab 0.996, unidic-mecab 2.1.2
spaCy 3.2.1	SudachiPy 0.6.2, SudachiDict-core 20211220
spacy-transformers 1.1.3	cl-tohoku/bert-base-japanese-v2, fugashi 1.1.1, unidic-lite 1.0.8
GPU RTX8000(48GB) x 2	CUDA 10.0, pytorch 1.9.1, transformers 4.12.5
CPU Xeon E5-2660 v3 x 2	DDR4-2666 32GB x 8, M.2 NVMe PCIe3.0x4 2TB, Pop!_OS 18.04 LTS