日本語のラベル付き依存構造解析の検討

田中 貴秋　　永田 昌明
NTT コミュニケーション科学基礎研究所
{tanaka.takaaki, nagata.masaaki}@lab.ntt.co.jp

1 はじめに

日本語の構文解析は，文節係り受け解析で行われることが多いが，直接得られる統語的な情報が乏しいことが問題になることがある．例えば，統計翻訳では，原言語を目的言語の語順に合わせる事前並べ替えを行うことで，翻訳結果の改善に効果があることが知られているが，日本語から外国語への翻訳では，述語と項の格関係や節の役割の情報が文節係り受け解析の情報から得られないため，構文解析だけでなく述語項構造解析が必要になる [1].

また，文節，あるいは複数の文節が結合した単位は，統語的機能を持つ単位と必ずしも一致しないため，文節係り受けに対して統語的な情報（依存関係ラベル）を付加することが難しい場合がある．並列構造を含む場合が典型的な例である．依存関係を結ぶ単位の制約を緩めるため，本稿では単語間の依存関係を考える．文節の制約を外すことによって，解析精度への影響が懸念されるが，森らは，単語間に依存構造を付与した大規模なコーパスを構築し，構文解析器の訓練データとして適用した結果として，90%以上の精度が得られることを報告している [2]．このコーパスには依存関係ラベルは付与されていないため，ラベル付きの依存構造解析の結果については検証する必要がある．

本稿では，日本語の構文解析で利用可能な統語的な情報を増やすため，句構造ツリーバンク [3] を依存構造に自動変換したデータを利用して，ラベル付き依存構造解析を行うことを試みた結果について述べる．ラベル付き依存構造解析は，従来の文節係り受け解析と比較して，(1) 依存関係の単位が文節から単語に変更される，(2) 依存関係のラベルが 2 種類（並列とそれ以外）から 35 種類に詳細化される，(3) 主辞の定義の仕方によって逆向きの係り受けが生じる（ただし，交差しない），という特徴がある．1 万文を使った比較的小規模な実験で，ラベル付き依存構造解析は，文節係り受け解析とほぼ同等の精度を保ちつつ，より詳細な統語情報を得られることが確かめられた．

2 日本語のラベル付き依存構造

2.1 単語の単位

ベースとなる単語の単位として，語の区切り，品詞に曖昧性，揺れの少ない BCCWJ コーパスの短単位 [4]（以下，単に短単位），すなわち UniDic[5] の見出し語を採用した．ただし，短単位は機能語等が細かい単位に分割される場合もあり，短単位の依存構造を直接扱うと，あまり意味のない単語間の関係を増やしてしまう可能性がある．そこで，依存関係を定義する単位としては，BCCCWJ コーパスの長単位（以下，単に長単位）を用いる．長単位は，構文的な機能に着目して規定した言語単位で，一つ以上の短単位から構成され，品詞に関しては文脈に即した曖昧性が解消したものが付与される．例えば,「結果」という語の短単位の品詞は「名詞-普通名詞-副詞可能」であるが,「これらの 結果 に基づき…」という文脈では「名詞-普通名詞-一般」となり,「その 結果, 様々な社会問題が発生し…」では「副詞」となる．また，長単位の導入により,「に対して」「かもしれない」等の機能表現の複合辞を，まとまった単位として扱うことができる．これにより，複合辞の内部構成を無視することができると同時に，内容語と機能語を適切に区別することが可能になるため（「対する」を内容語として扱わない），単語間の依存関係ラベルを付与するのに都合が良いと考える．一方で，長単位の内部構成を無視することにより，依存構造を作る場合に問題を生じる可能性もあるが（「部分的 ‖ な ‖ 学校五日制導入以来」[1]のような名詞連続で構成される長単位を含む場合など），本稿では扱わず後の分析課題とする．

2.2 主な依存関係ラベル

日本語の依存構造に使用する依存関係ラベルは，Stanford typed dependency (SD) [6] を参考にして 35 種類を定義した．SD は単語間の統語的な関係を 4-50 程度のラベルにより表現する．SD の多言語への拡張として，言語横断的な構文構造の表現を目指した Universal dependency [2] [7] があるが，本稿は，日本語単一言語内での統語情報の詳細化を目指し，SD を出発点として考える[3].

単語間の依存構造への依存関係ラベルの導入により，付加される主な統語的情報には以下のようなものがある．基本的には，依存構造に変換するのに使用した句構造ツリーバンク [3] の持っている情報をそのまま移植している．

[1]「‖」は長単位の境界を示す.
[2]https://universaldependencies.github.io/docs/
[3]Universal dependency との相互変換が行えるように検討を進める予定である.

格関係 (必須格): (対応する句構造の非終端記号: -SBJ, -OBJ, -OB2) 述語と項の関係．主格 <*nsubj*>，対格 <*dobj*>，与格 <*iobj*> を定義する．受動態，使役態等の場合も格交替後の格で表示する．

格関係 (随意格): (-TMP, -LOC) 述語と項の関係．時間格 <*tmod*>，場所格 <*lmod*> を定義する．

関係節: (IP-REL_sbj, IP-REL_obj, IP-REL_ob2) 関係節と主名詞との関係（内の関係）．_ 以下のラベルで空所になっている格も表示する <*rcmod_nsubj*>, <*rcmod_dobj*>, <*rcmod_iobj*>.

内容節，補充節: (IP-ADN) 上記関係節以外の連体修飾節と被修飾名詞との関係 (外の関係)<*ncmod*>.

副詞節 (IP-ADV) 副詞節と主節の間の関係 <*advcl*>.

並列: (-COORD) 語，句，節など各要素の並列関係 <*conj*>.

同格: (-APPOS) 名詞 (句，節) 間の同格関係 <*appos*>.

2.3 句構造から依存構造への変換

依存構造に変換する元となるツリーバンクとして，京大コーパス [8] に句構造を付与した日本語句構造ツリーバンク [3] を使った．このツリーバンクの句構造は完全な二分木で構成されている．各部分木に対して，主辞決定のルールと依存関係ラベル決定ルールを順に適用することにより，依存構造に変換した．

日本語は主辞後置型の言語であり，基本的には，依存関係のある語のうち右側の語が主辞になると考えられ，以下のような原則で依存構造を作る．

- 後置詞句 (PP) は，助詞（格助詞等）を主辞とする
- 副助詞，句読点，閉じ括弧類は左側の要素を主辞とする
- 並列句では，最右の要素を主辞とする

ただし，単純に右側の語から左側に順に依存関係を作ると，述部の構造において付属語の階層が深くなり，連体修飾節などにおいて内容語（動詞と名詞など）の関係が捉えづらくなり，依存構造ラベルの精度に影響する可能性がある．そこで，述語の扱いの違いよって，3 つのタイプの依存構造を考えた．

主辞後置型 1 (HF_1) 述部の最左要素（内容語）と格要素となる後置詞句 (PP) の間で先に構造を作る句構造を想定し，副助詞，句読点など一部を除いて，各構造で右側を主辞とする．格要素となる後置詞句と述部は，後置詞句の最右要素（助詞）と述部の最左要素（内容語）に依存構造を作る．

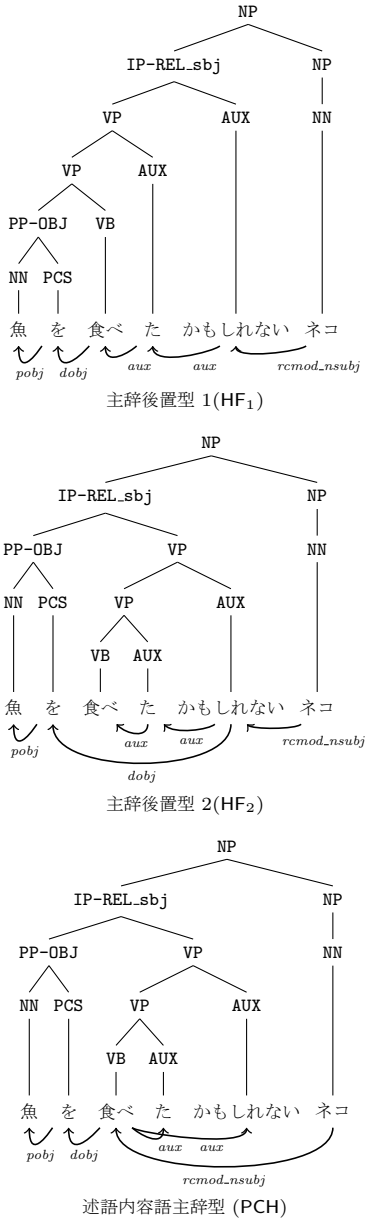

図 1: 連体修飾節の句構造と依存構造への変換例

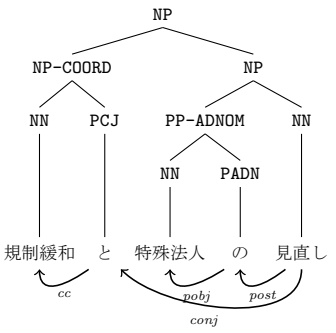

図 2: 並列句の句構造と依存構造への変換例

図 3: 解析器の構成

主辞後置型 2 (HF_2) 接続助詞を除いて述部の文節相当の単位で先に構造を作る句構造を想定し，副助詞，句読点など一部を除いて，各構造で右側を主辞とする．格要素となる後置詞句と述部は，後置詞句の最右要素（助詞）と述部の接続助詞を除く最右要素（機能語）に依存構造を作る．

述部内容語主辞型 (PCH) 述部の文節相当の単位で先に構造を作る句構造を想定する．述部において最左要素（内容語）を主辞とし，各機能語は内容語の付属部とする．

句構造と各タイプの依存構造との対応関係の例を図 1 に，並列構造を含む依存構造の例を図 2 に示す[4]．HF_2, PCH では，主辞が左側になる逆向きの依存関係が生じるが，3 タイプの依存構造ともに交差は生じない．

3 解析器の構成

本稿で想定する解析器の構成を図 3 に示す．ベースとなる形態素解析は，MeCab [5] に，解析辞書 unidic-mecab [6]を使用して行う．この結果に対して，長単位解析器 Comainu [9] を使用して，長単位へのチャンキングを行う．依存構造はこの長単位の単語間に対して考えるが，依存構造解析器に使用する素性には，短単位，長単位の両方の情報を使用する．本稿では，依存構造解析器として，英語の SD で実績のある MaltParser[10] を使用した．

4 構文解析の評価

句構造ツリーバンクをラベル付き依存構造に変換したコーパスを訓練データとして MaltParser の解析モデルを構築し，構文解析精度の評価を行った．京大コーパスの一部を対象として構築した句構造 10,000 文のうち，8,000 文を訓練セット，2,000 文を評価セットとした．使用したツリーバンクは IPA 品詞体系で構築されているため，別に作成した UniDic 形態素解析データをもとに長単位の句構造に自動で一旦変換した後，依存構造に変換した[7]．依存構造解析器で使用する素性は，表 1 に

長単位	
l_FORM	書字形
l_LEMMA	語彙素
l_UPOS	短単位品詞
l_INFTYPE	活用型
l_INFFORM	活用形
l_CPOS	句構造前終端記号
l_SEMCLASS	一般名詞意味属性
l_PNCLASS	固有名詞意味属性
短単位	
s_FORM_R	書字形 (最右)
s_FORM_L	書字形 (最左)
s_LEMMA_R	語彙素 (最右)
s_LEMMA_L	語彙素 (最左)
s_UPOS_R	長単位品詞 (最右)
s_CPOS_R	句構造前終端記号 (最右)
s_SEMCLASS_R	一般名詞意味属性 (最右)
s_PNCLASS_R	固有名詞意味属性 (最右)

表 1: 素性に使用した使用した単語属性（短単位中の_L, _R は，それぞれ元の長単位を構成する短単位列中での最左，最右の短単位を示す.）

示すように長単位ベースの単語属性，短単位ベースの単語属性を併用して定義する．依存関係を持つ単語は長単位であるので，短単位に関する属性は，長単位を構成する最左または最右の語に関する属性を用いる．例えば，「調査 | し」[8]の長単位に対して，s_FORM_L, s_LEMMA_L は，ともに「調査」，s_FORM_R, s_LEMMA_R は，それぞれ「し」,「する」となる．また，品詞情報は，UniDic の品詞を変換することにより，句構造ツリーバンクの前終端記号として用いた Penn Treebank 型の 26 の粗い品詞シンボル (e.g. NN, PCS) を併用する [3]．他に，各単語に対して日本語彙大系 [11] の一般名詞意味属性，固有名詞意味属性を付与した．MaltParser は shift-reduce parser であるので，これらの属性を使ってスタック, 入力バッファ, 部分木に関する素性を定義した．

MaltParser の解析アルゴリズムは，Stack アルゴリズム (projective)，学習器は LIBLINEAR を使用した．各依存構造のタイプごとの結果を，表 2 に示す．表には，同一の文のセットに対して，文節係り受けの構文解析器の CaboCha[12] で文節区切り，係り受けを学習して解析モデルを構築し，解析結果を単語依存関係に換算した結果も併記している[9]．

全体としての精度に関して，変換した依存構造のタイプによる差は，ラベル付きの精度 (LAS)，ラベルなしの精度 (UAS) ともにほとんど見られない．また，ラベル付き依存構造解析器（単語単位），文節係り受け解析器の結果もほとんど同等と考えられる．特に，LAS の結果はラベル付き依存構造解析器で用いたラベル数が 35, 文節係り受けは，並列 (P) とそれ以外 (P) の 2 種類であることを考えると，比較的高い結果を示していると考えられる．ただし，今回使用した学習データがあまり大きくないため，規模を大きくした場合の結果も検証する必要がある．

[4]使用した実際のツリーバンクは，文節相当で述部の構造を作る形 (HF_2, PCH) になっている．
[5]https://code.google.com/p/mecab/
[6]http://sourceforge.jp/projects/unidic/
[7]長単位解析は 自動解析結果による誤りが含まれている．
[8]「|」は短単位の境界を示す．
[9]ただし，IPA の品詞体系により解析した結果を用いているので，厳密な意味での比較はできない．

	UAS	LAS
HF_1	93.7	89.1
HF_2	94.1	89.4
PCH	93.2	89.0
CaboCha	92.7	90.1[a]

[a]ラベルは並列 P とそれ以外 D の 2 種類である

表 2: 構文解析結果

	F_1 score			
依存関係ラベル	HF_1	HF_2	PCH	(句構造)[a]
nsubj	79.0	81.2	79.9	75.1
dobj	91.1	90.5	91.4	77.7
iobj	80.8	77.7	80.4	67.3
tmod	49.7	53.2	52.3	46.5
lmod	45.1	46.0	43.4	18.2
rcmod_nsubj	68.7	68.8	71.2	51.1
rcmod_dobj	29.9	33.8	37.1	24.9
rcmod_iobj	26.7	28.6	26.2	21.7
ncmod	81.6	81.5	81.5	53.1
advcl	65.1	66.5	62.3	66.4
conj	69.3	68.9	69.1	61.6
appos	49.5	53.2	51.0	42.8

[a]依存構造の実験とデータセットが異なる

表 3: 構文解析結果 (依存構造ラベル別)

表 3 は，各依存関係ラベルごとの F 値を示している．参考として，変換元となった句構造を学習データとして用い，Berkeley Parser[13] で解析した際の対応する非終端記号ごとの解析精度を示す（ただし，今回とデータセットの構成が異なる [3]）．格関係のラベルのうち，*nsubj*, *dobj*, *iobj* は，格助詞が明示されていることが多いため，比較的高い結果が出ている．また，関係節ラベル間 *rcmod_∗* の判別は，かなり低い結果となっている．関係節の空所の格を区別するためには，述語と主名詞の意味的関係を捉える必要があるため，後段の格解析，意味役割付与と合わせて行う方が良いと考えられる．

依存構造解析の結果の傾向としては，句構造解析での結果とほぼ同様であるが，全体に依存構造解析の方が高い精度が得られている．この要因の一つとして，依存構造に変換することにより，元々の句構造の階層構造が除去され，依存関係にある語の語彙情報や，複数の依存関係ラベルの組合せが素性に反映し易くなったためでないかと考えられる．

5 おわりに

句構造ツリーバンクから変換した依存構造を利用して，日本語のラベル付き依存構造解析を行い，小規模なデータセットにおいて，文節係り受けと同等な精度が得られることを示した．今回取り扱わなかったが、長単位を依存構造の単位とする場合の課題（主に複合名詞や機能語の複合語の内部構造が扱えない点）の分析をすすめ，統計翻訳の並び替えへの適用や後段の述語項構造解析，省略解析と組み合わせた解析について検討を進める予定である．

参考文献

[1] Sho Hoshino, Yusuke Miyao, Katsuhito Sudoh and Masaaki Nagata: Two-stage pre-ordering for Japanese-to-English statistical machine translation. In *Proc. of IJCNLP 2013*, pp.1062 – 1066 (2013)

[2] Shunsuke Mori, Hideki Ogura and Teturo Sasada: A Japanese word dependency corpus. In *Proc. of LREC 2014*, pp. 753 – 758 (2014)

[3] 田中貴秋, 永田昌明, 松崎拓也, 宮尾祐介, 植松すみれ: 統語情報と意味情報を統合した日本語句構造ツリーバンクの構築. 言語処理学会第 20 回年次大会予稿集, pp.737–740 (2014)

[4] Kikuo Maekawa, Makoto Yamazaki, Toshinobu Ogiso, Takehiko Maruyama, Hideki Ogura, Wakako Kashino, Hanae Koiso, Masaya Yamaguchi, Makiro Tanaka and Yasuharu Den: Balanced Corpus of Contemporary Written Japanese. Language Resources and Evaluation, Vol. 48, pp. 345 – 371 (2014)

[5] Yasuharu Den, Jupei Nakamura, Toshinobu Ogiso and Hideki Ogura: A proper approach to Japanese morphological analysis: Dictionary, model and evaluation. In *Proc. of LREC 2008* (2008)

[6] Marie-Catherine de Marneffe and Christopher D. Manning: The Stanford typed dependencies representation. In *Proc. of COLING 2008 Workshop on Cross-framework and Cross-domain Parser Evaluation* (2008)

[7] 金山博, 宮尾祐介, 田中貴秋, 森信介, 浅原正幸, 植松すみれ: 日本語 Universal Dependencies の試案. 言語処理学会第 21 回年次大会予稿集 (2015)

[8] Sadao Kurohashi and Makoto Nagao: Building a Japanese parsed corpus – while improving the parsing system. In Abeille (ed.), Treebanks: Building and using parsed corpora, Chap. 14, pp. 249–260. Kluwer Academic Publishers (2003)

[9] 小澤俊介, 内元清貴, 伝康晴: BCCWJ に基づく長単位解析ツール Comainu. 言語処理学会第 20 回年次大会予稿集, pp.582-585 (2014)

[10] Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov and Erwin Marsi: MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(2), 95-135 (2007)

[11] 池原悟, 宮崎正弘, 白井諭, 横尾昭男, 中岩浩巳, 小倉健太郎, 大山芳史, 林良彦:日本語語彙大系, 岩波書店, (1997)

[12] Taku Kudo and Yuji Matsumoto: Japanese dependency analysis using cascaded chunking. In *Proc. of CoNLL 2002*, Vol. 20, pp. 1–7 (2002)

[13] Slav Petrov, Leon Barrett, Romain Thibaux and Dan Klein: Learning accurate, compact, and interpretable tree annotation. In *Proc of COLING-ACL 2006*, pp. 433-440 (2006)