アノテーション方式とコーパスの特色

吉本 啓
東北大学
kei@compling.jp

1 はじめに

本発表では、文法研究を目的とするテクストデータのピンポイントの検索および意味解析情報の抽出という目標を達成するために本研究において編み出されたアノテーション方式について解説する。

次節では、日本語の使用の実情に適合させるために作り出されたアノテーションの原則を具体例とともに解説する。次に、コーパスをインターネットを介して高速、効率的に利用するための XML エンコーディングについて述べる。また、Emacs を利用したアノテーション支援ツールについて紹介する。最後に、本アノテーション法の持つ特色と意義について述べる。

2 アノテーションの方法と具体例

2.1 構築方法

図 1 に本コーパス開発の過程を示す。テクストは①で、MeCab,UNIDIC および Comainu を使用して形態素解析に掛けられる。名詞句は長単位、述語句は短単位にもとづく解析が原則である. その結果は②で、Probabilistic Context-Free Grammar にもとづく統語解析を受ける。その解析結果は多くの誤りを含むので、人手による修正が必要である。また、修正結果は統語解析プログラムへとフィードバックされる。修正を経た統語解析結果は、Awk/Perl で作成した構造変換プログラム (③) により、Scope Control Theory (SCT; Butler 2015) にもとづく中間表示に変換される。これが Standard ML で作成した意味解析システム (SCT のインプリメンテーション、④) に入力され、論理意味表示を出力する。

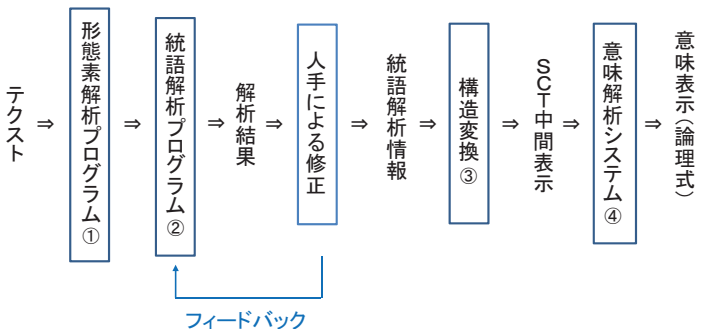

図 1: コーパス開発の過程

2.2 ラベル

現在構築中の日本語ツリーバンクでは、タグ付け基準の客観性・一貫性、日本語使用の実情、および意味表示からの要請、という時には互いに矛盾も生じるそれぞれの条件を最大限満たし、バランスの取れたコーパス開発の方針の確立を目指している。

語彙ラベル (品詞情報) のタギングは、上記のようにペン通時コーパス (Santorini 2010) のものを基本とし、益岡・田窪 (1992) を参考にしつつ、意味情報付きコーパスの開発という独自の目的に適うように行っている。MeCab を使って自動形態素解析した結果を人手で修正している。語彙ラベルとしては、表 1 の 27 種類 (記号を除く) を使用する。

句に対しては、表 2 の 13 種類のラベルが付加される。このうち、CP, IP、NP および PP は機能を表すラベルとともに用いることが出来る。CP, IP および PP は機能ラベルが必須である。

ADJI	い-形容詞	NPR	固有名詞
ADJN	な-形容詞	NUMCL	助数詞
ADJT	たる-形容詞	P	助詞
ADV	副詞	PASS	受動助動詞
AX	助動詞	PRO	代名詞
AXD	過去テンス標識	Q	量化詞
CARD	数詞	VB	動詞 (語幹)
CONJ	並列接続詞	VBO	軽動詞
D	指示限定詞	VB2	補助動詞
FW	外来語	WADV	疑問副詞
INTJ	間投詞	WCARD	疑問数詞
MD	モーダル助動詞	WD	疑問限定詞
N	名詞	WPRO	疑問代名詞
NEG	否定辞		

表 1: 品詞タグ

ADJP	形容詞句	NML	中間名詞節
ADVP	副詞句	NP	名詞句
CONJP	接続詞句	NUMCLP	助数詞句
CP	従属節	PP	後置詞句
FRAG	断片	PRN	カッコ挿入句
INTJP	間投詞句	QP	量化詞句
IP	節		

表 2: 統語タグ

2.3 重要な原則

NPCMJ のアノテーション方針のうち、独特かつ重要なものについて説明する。

(i) いくつかの単語が緊密に連結して 1 つの機能語として働くものは、1 つの助詞 (P) として扱う。

これは、最終目的である文意味解析の便宜のためである。この中には、「として」「について」「に対して/対する」「に関して/関する」等が含まれる。このうち、「として」「について」には助詞プラス動詞テ形としての用法もあり、構造的にあいまいなものとして取り扱う。また、通常形式名詞とされる「ため」「おかげ」「せい」「あまり」についても、「のために」のように 1 つの機能語に相当する用例については、1 語の P とする。

(ii) いくつかの単語が緊密に連結し 1 つのモーダル的機能を果たすものは、1 つの助動詞 (MD) とする。

これも、文意味解析の便宜のために行う。例えば、「なければならない」は 1 個のモーダル助動詞 MD とする。

(iii) 後置詞句 (PP) が文中で主語や目的語として機能する場合、その直後に NP-SBJ, NP-OB1 または NP-OB2 を付加して、その文法機能を明示する。

これには、係助詞「は」や副助詞が付加されて格が明示されない場合を含む。しかし、格助詞「が」、「を」や「に」を伴う場合でも、格助詞により表示される文法役割があいまいなため、この方法によって格情報を明示する。会話文などでこれらの格助詞が省略された名詞句についても同様の表示を行う。例は「イントロダクション」の例文 (2b) の 7 行目および (3b) の 9 行目に示されている。

(iv) 関係節が修飾する名詞句において、主名詞が関係節の中で文法役割を果たす場合は、関係節内に空所に相当するノードを与えて文法役割を明示する。

(v) 主語または目的語が動詞の必須格として求められるにもかかわらず文中で表現されていない場合の多くについて、それらをゼロ代名詞として明示する。

ゼロ代名詞のタギングを行うのは、依存関係の表示および述語-項関係の再構成に必要だからである。無主語文の主語のゼロ代名詞表示は必要無い。以下の (1a) の最初の行に、目的語をゼロ代名詞として表示した例を示す。(1b) の意味表示では、文脈中の先行詞と同定されるべき値として扱われている。

(1) a.
```
(IP-MAT (NP-OB1 *pro*)
        (PP (PP (NP (WPRO 誰))
                (P-OPTR か))
            (P-CASE が))
        (NP-SBJ *が*)
        (VB 助ける)
        (MD だろう)
        (PU 。))
```

b. $\exists yx(\text{pro:}y \ = \ ? \ \wedge \ \text{だろう}(\exists e_1 \, \text{助ける}(e_1, \ x, \ y)))$

主語や目的語が明示されなていなくても、それと同一指示の名詞句が文中に存在してコントロール関係にある場合は SCT のスコープ操作によってコントロール関係が意味論のレベルで補完されるため、ゼロ代名詞としてのタギングは行わない。以下に例を示す。

(2) a.
```
(IP-MAT (NP-SBJ *pro*)
        (PP (IP-TE (ADVP (ADV よく))
                   (VB 考え)
                   (P-CONJ て))
            (P-CASE から))
        (PU 、)
        (VB ご返事)
        (VBO し)
        (AX ます)
        (PU 。))
```

b. $\exists x e_1 p_1($
$\text{pro:}x \ = \ ? \ \wedge$
$\text{fact}(p_1, \ \exists e_2(\text{考え}(e_2, \ x) \ \wedge \ \text{よく}(e_2))) \ \wedge$
$\text{ご返事\_し\_ます}(e_1, \ x) \ \wedge \ \text{から}(e_1) \ = \ p_1)$

(vi) 例外的な場合を除き、インデックスは使用しない。

長距離依存のような複雑な構文でもインデックスが不要であることは SCT の大きな特徴であり、これによって、本研究で目指しているツリーバンク構築の作業量が著しく軽減される。また、表層的な自動統語解析結果を SCT システムへの入力とし、述語論理式出力までのすべての過程を自動化することも視野に入れることが出来る。例外となるのは、外置 (extraposition)、数量詞遊離 (floating quantifier)、主要部内在型関係節のいずれかの構文で、意味処理上の要請から、語句をその実際の位置以外の場所と関係づける必要が生じる場合である。

3 XMLによるエンコーディング

Penn Treebank をはじめとするこれまでのツリーバンクでは文の統語解析情報をカッコの埋め込みによって表示してきたが、このことは本プロジェクトの準備段階における欅ツリーバンク・プロトタイプの構築においても順守されてきた。カッコの埋め込みによる解析木の表示は人手による修正にとって便利であり、また木の編集のための tregex (Levy and Andrew 2006) などのツールもそろっていることから、本プロジェクトでも引き続きコーパス開発のために利用する。

しかし、解析木のカッコ表示は検索の効率性に問題があり、しかも木の中のノードに持たせる情報の拡張が容易でない。これに対し、XML で記述したデータベースに対しては効率的に検索を行う技術が確立されており、しかも各ノードに対して従来のように単一の文字列だけを与えるのでなく、レンマなど様々な種類の情報を持たせることが可能になる。そこで本プロジェクトでは、カッコ表示方式により構築したコーパスを XML のマークアップ方式に変換し、さらに必要に応じ様々な情報を付加して拡張していくことにする。また、最近になって、XPath (XML ドキュメントの特定部分の指定に用いられる言語で、木構造に対するクエリとして使用される) XSLT や XQuery など、ウェブインタフェースのための一般的な技術が普及しており、さらに Dact (Noord, et al. 2013) や PaQu (Odijk 2015) のような統語アノテーションを操作したり検索するためのツールも開発されてきている。XML はこれらの技術やツールとも適合しており、したがって XML マークアップの利用によって本プロジェクトでもそれらにもとづくウェブインタフェースの構築が可能になる。

本プロジェクトで採用する 'Alpino' XML による解析木のエンコーディングについて以下に述べる。これは Noord et al. (2013) によるオランダ語ツリーバンクの方式にもとづいている。木構造は XML の要素である node の再帰的な埋め込みによって示される。各ノード中の XML-素性には以下のものがあり、これらに対して様々な値が与えられることによって文法情報が記述される。

`cat` 非終端ノードの統語的カテゴリー
`pt` 終端ノードの品詞タグ
`word` 単語または空要素
`begin` 表層文字列中の当該ノードが始まる位置
`end` 表層文字列中の当該ノードが終了する位置

例として、文「輪を回す」に相当する XML マークアップを挙げる。

```
<alpino_ds id="2_textbook_kisonihongo;page_13" version="1.3">
  <node cat="ip-mat" id="1" begin="0" end="5">
    <node cat="np-sbj" id="2" begin="0" end="1">
      <node pt="zero" word="*pro*" id="3" begin="0" end="1"/>
    </node>
    <node cat="pp-ob1" id="4" begin="1" end="3">
      <node cat="np" id="5" begin="1" end="2">
        <node pt="n" word="輪" id="6" begin="1" end="2"/>
      </node>
      <node pt="p-case" word="を" id="7" begin="2" end="3"/>
    </node>
    <node pt="vb" word="回す" id="8" begin="3" end="4"/>
    <node pt="pu" word="。" id="9" begin="4" end="5"/>
  </node>
  <sentence>*pro* 輪を回す。</sentence>
</alpino_ds>
```

XML-素性のうち、begin と end によって語句の位置がエンコードされている。例えば、x ノードの直後に y ノードが後続することは、x の end-素性の値が y の begin-素性の値に等しいことにより示すことが出来る。

4 Emacs けやきモードによるアノテーション作業

上述のように、NPCMJ プロジェクトでは、ペン通時コーパスを元にしたアノテーション基準に従って、プレーンテキスト形式で統語構造のアノテーションを行っている。PCFG パーザーの出力結果を目でチェックしてエラーを直すことが、人間のアノテーターの行う作業の中心となる。以下では、アノテーション作業の実際を、使用しているツールの解説を交えて説明する。

現在、本プロジェクトでは、**Emacs けやきモード**と呼ばれる、Emacs のマクロとして実装したアノテーション支援ツール (窪田悠介作成) を主に使用して、パーザーのエラーの修正を行っている[1]。エラーは、(i) ノードのアッタチメントの間違い、(ii) ノードラベルの間違い、(iii) 単語の切り分けの間違いに大別される。

けやきモードの主な機能は、編集操作の補助と外部プログラムとの連繋に大別される。編集補助機能のうち、最も重要なものはノードのアタッチメントの修正である。図 2 に、修正作業の画面の例を示す。

アタッチメントの修正は、マウスによるメニュー操作により行うことができる。アノテーターは、ブラケット修正の煩雑な編集操作を直接手で行うことなく、「当該の構成素をどの述語にかかる要素として再分析すべきか」という言語学な問題に集中することで必要な操作を完了することができる。

エラー修正と、コーパス全体の一貫性のチェックのため、アノテーターは、すでにエラー修正作業が終わったファイルを参照しながら作業を行っている。

図 3 に、けやきモードからコマンドライン・ツールを呼び出して「のような」という文字列のタグ付けを修正済みコーパスから検索している画面の例を示す。

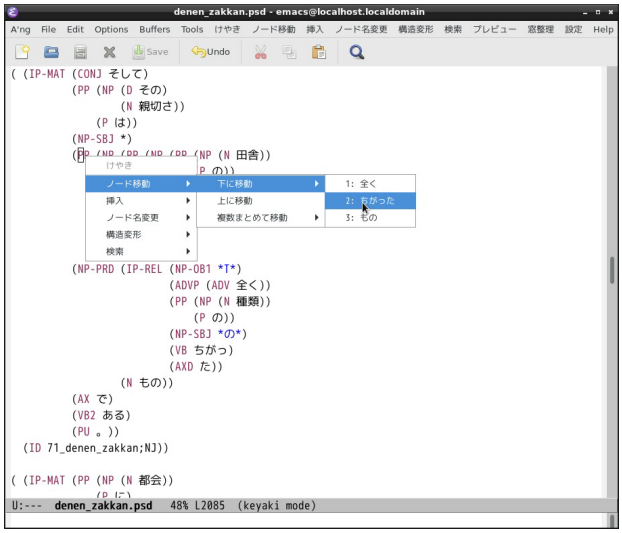

図 2: 修正作業の画面

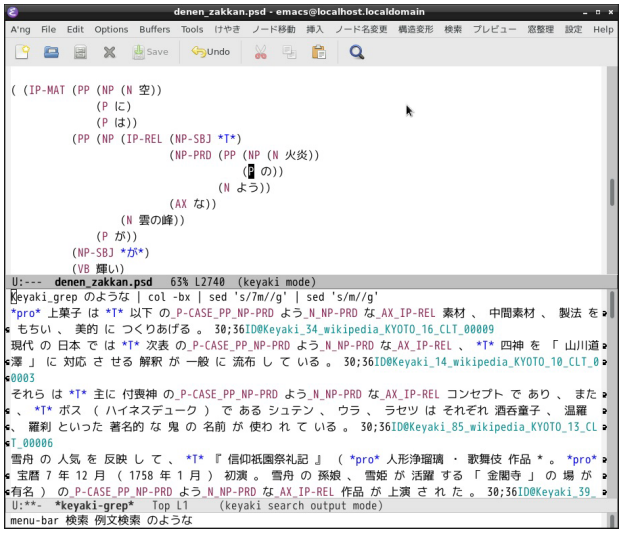

図 3: 修正済みコーパスからの検索

[1]Emacs のマクロとしてコーパス開発用のインターフェイスを作る開発手法は様々なメリットがあり、幅広く応用可能であると考えられる。この点に関して、より詳しくは 窪田 (2016) を参照のこと。

5　結論–特色と意義

現状の一般のツリーバンクのアノテーションは、表層的な統語解析情報の提供にとどまる。ツリーバンクも含め、現在世界に存在するコーパスは、対象とする言語にかかわらず語句間の**共起 (co-occurrence)** に関する手掛かりを与えるにすぎず、それらの間の**依存関係 (dependency)** について正確な情報を与えることは出来ない。このようなコーパスを使っても、研究者は自分のほしい文法情報を持つデータをおおまかに絞り込むことが出来るだけで、最終的には手と目を使ってデータを選り分けることが必要になる。それだけでなく、検索結果が必要なデータを取りこぼしている可能性すらある。

これに対して、NPCMJ では、有界依存構文 (unbounded dependency) のような複雑な構文に関しても、その依存関係を論理意味表示の形で把握している。たとえば、「イントロダクション」中の例文 (2a) において関係節中の動詞「撮った」の目的語が主名詞「写真」であることは、論理式 (2c) において共通の変項 x が述語「写真」および「撮っ」の項として表れることによって示されている。このことにもとづいて、表層的な統語解析情報としてはタグ付けされていない複雑な構文中の依存関係についても抽出することができ、研究者にとって必要な文法情報をピンポイントで提供することを可能にするのである。

参考文献

Butler, Alastair. (2015) *Linguistic Expressions and Semantic Processing: A Practical Approach.* Springer.

窪田悠介「統語構造アノテーション支援ツールの開発」、草稿、筑波大学. 2016.

Levy, Roger, and Galen Andrew. Tregex and tsurgeon: tools for querying and manipulating tree data structure, 5th International Conference on Language Resources and Evaluation, 2006.

益岡隆志・田窪行則 (1992)『基礎日本語文法—改訂版—』 くろしお出版。

Noord, Gertjan van, Gosse Bouma, FrankVan Eynde, Daniel deKok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. Large scale syntactic annotation of written Dutch: Lassy. In: P. Spyns and J. Odijk, eds., *Essential Speech and Language Technology for Dutch: Resources, Tools and Applications*, pages 147–164. Springer. 2013.

Odijk, Jan. Linguistic Research with PaQu. *Computational Linguistics in the Netherlands Journal* 5: 3–14. 2015.

Santorini, B. Annotation Manual for the Penn Historical Corpora and the PCEEC (Release 2). Tech. rep., Dep. of Computer and Information Science, University of Pennsylvania. 2010.