1A1-01

正しい構文木の選択を支援する構文木付きコーパス作成ツール

A syntactic annotation tool with user navigation

岡崎 篤　白井清昭　徳永健伸　田中穂積
Okazaki Atsushi　Shirai Kiyoaki　Tokunaga Takenobu　Tanaka Hozumi

東京工業大学 大学院情報理工学研究科
Department of Computer Science, Tokyo Institute of Technology

Annotated corpora play crucial role in corpus-based natural language processing. In order to reduce human labor in constructing large annotated corpora, versatile annotation tools are indispensable. This paper describes a annotation tool for constructing syntactically annotated corpora. The tool receives a set of syntactic trees given as a parsing result, and presents candidate trees to users. User can give various constraints to the presented trees, and each constraint works as a filter to filter out irrelevant trees. This tool also has a function to navigate users to give effective constraints. An effectiveness of a constraint is estimated based on accumulated constraints users gave in the past. The experiments showed that the navigation function is promising for reducing human labor in annotating corpora.

1. はじめに

様々な自然言語処理に利用される言語資源の一つに、実際に使われている例文を収集したコーパスがある [1]。その中でも、形態素区切り, 品詞, 係り受けの情報が付与された構文木付きコーパスは、自然言語処理システムの構築や評価に用いられる有用な言語資源である。しかしながら、例文に対して正しい構文木を付与するためには、大変な労力が必要である。したがって、構文木付きコーパスの作成を支援し、その労力を軽減させることは、重要な研究課題の一つである。

このような背景から、本論文では、正しい構文木を付与する作業者の負担を軽減させる機能を持つ構文木付きコーパス作成支援ツールについて述べる。

2. 構文木付きコーパス作成ツール

2.1 概要

本論文では、構文木付きコーパスを、以下の手続きによって作成することを考える。

1. コーパスから文を取り出す
2. その文を構文解析し、構文木の候補の集合を得る
3. 得られた構文木の集合の中から正しい構文木を選ぶ
4. 選んだ構文木をコーパスに与える

構文木の集合から正しい構文木を選ぶ作業 2 を行うのが、本論文で述べるツールの役目である。その概要を図 1 に示す。ツールへの入力は、構文解析の結果得られる構文木の候補の集合である。具体的には、形態素・構文解析ツール MSLR パーザ [2] が出力する圧縮統語森である。ツールは、圧縮統語森中から構文木を一つ取り出し、それを視覚的に表示することができる。

多数ある構文木の候補の中から正しい構文木を選ぶ際、一つ一つ構文木の構造を見て正しいものを探すのは時間を要する。そこで、正しい構文木が満たすべき制約をユーザが教示して、構文木の候補の中でその制約を満たす構文木だけを残す操作をツールで行なう。これを何回か行なって候補を絞りこんだ上で、正しい構文木をファイルに保存する。ユーザが教示できる制約の種類については、2.2 節で説明する。

連絡先: 東京工業大学 大学院情報理工学研究科計算工学専攻 徳永研究室, 〒 152-8552 東京都目黒区大岡山 2-12-1, E-mail : okazaki@cl.cs.titech.ac.jp

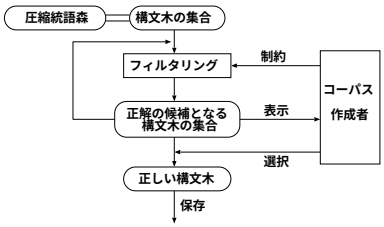

図 1: コーパス作成の流れ

図 2 に本ツールの表示画面を示す。メニューバーの下にある左右の矢印ボタンによって、制約を満たす構文木を順番にスライド表示させることができる。これにより、候補数が少なくなれば正しい構文木を選択するのが容易になる。左右の矢印ボタンの間にある “2850” は全構文木数を表し、“2/3” は絞り込まれた結果残っている構文木 3 つの中で現在表示している木が 2 番目のものであることを示している。

その他、ユーザが与えた教示の Undo/Redo、教示した制約を記録したログファイルの保存/読み込み、構文木の各部分に曖昧性が存在するかどうかを色分けして表示する機能など、ツールの利便性を向上させるための機能をいくつか実装した。

2.2 ユーザの教示

ユーザは、正しい構文木が満たすべき制約として、以下の 5 種類の制約を教示することができる。

- 正しい形態素区切りを教示する
- 正しい品詞を教示する
- 正しい非終端記号を教示する
- 正しい係り受けを教示する
- 間違い箇所を直接教示する

本ツールでは、教示する順序についての制限はない。しかし、基本的には、まず形態素区切りを教示し、次に品詞を教示

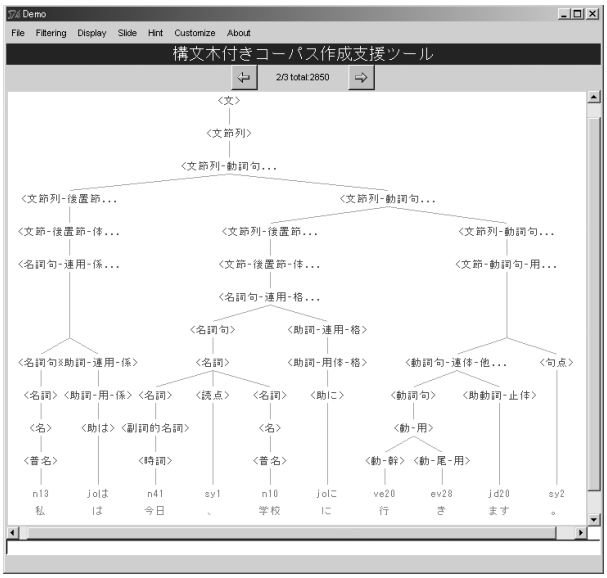

図 2: ツールの表示画面

し、最後に係り受けなどの構造を教示するというように、構文木の葉から根に向かって制約を与えることを想定している。以下、それぞれの教示の方法を詳しく説明する。

形態素区切りの教示
現在表示されている構文木の形態素区切りが間違っている場合、図 3 のようなメニューによって形態素区切りを修正することができる。その結果、指定したものと同じ形態素区切りを持つ構文木だけが残される。
具体的には、形態素区切りの位置を左右にずらしたり、隣接する 2 つの形態素を 1 つに結合したり、1 つの形態素を分割したりする機能を持つ。

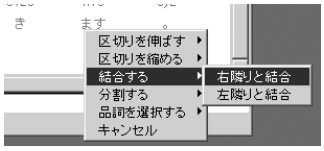

図 3: 正しい形態素区切りを教示するメニュー

品詞の教示
形態素区切りが正しいとしても、例えば「ます」という形態素に助動詞と普通名詞があるように、品詞に関して曖昧性が存在する場合がある。本ツールでは、正しい品詞を図 4 のようなメニューで選択することで、指定した品詞を持つ構文木だけを残す機能を持つ。

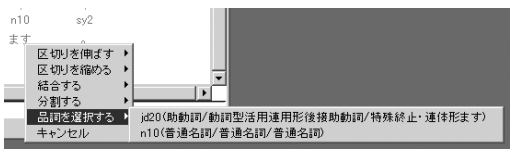

図 4: 品詞を選択するメニュー

非終端記号の教示
構文木を木の形で図示した時に、節点として文法の非終端記号が表示される。文中のある範囲の文字列に対して、その文字列を支配する非終端記号を教示することができる。ただし、非終端記号が単一の子を持つために同一の文字列を支配する節点が複数存在する場合、葉に最も近い節点を対象とする。例えば、図 5 のようなメニューで連体修飾と連用修飾のどちらであるかを指定することにより、指定された以外の非終端記号を持つ構文木がすべて候補から取り除かれる。

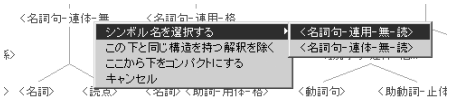

図 5: 非終端記号を選択するメニュー

係り受けの教示
構文解析の曖昧性の多くは、係り受けの曖昧性に起因する。本ツールでは、文中のある範囲に対して、その係り先を教示することができる。係り元は、一つの形態素であっても、複数の形態素を含むような範囲であっても構わない。係り受けについて制約を与えると、その制約に矛盾するような構文木がすべて取り除かれる。例えば「私は今日、学校に行きます。」という文で、「今日、」が「行きます。」に係るという教示を行なうと、「今日、学校に行きます。」を支配する非終端記号が存在し、かつ「学校に行きます。」を支配する非終端記号が存在しなければならない。したがって、図 6 のような構造を持つ構文木だけが残り、図 7 のような構文木 (「今日、」が「学校に」に係っている) は除かれる。

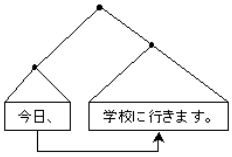

図 6: 制約を満たす構造

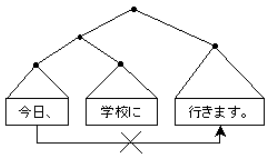

図 7: 制約を満たさない例

さらに、ある範囲に対して、非終端記号や係り先に制約を与えたということは、暗黙のうちにその範囲を支配する非終端記号が必ず存在することを意味すると考え、その範囲を支配する非終端記号が存在しない構文木は除かれる。

間違い箇所の教示
人間が見ると明らかに間違っていると分かる構造を持つ部分木が表示されている時、その部分木が間違っていることを直接教示する方法を用意した。具体的には、ある非終端記号に対して、その直下の解釈が間違っていることを教示すると、その部分木を含むような構文木すべてが取り除かれる。例えば、「今日、学校」を複合名詞と解釈している図 8 のような構文木を簡単に取り除くことができる。

3. 構文木付きコーパスの作成支援

2 節で述べた機能に加えて、どの箇所について教示を行なうとより効果的に候補を絞り込むことができるか（曖昧性を減少できるか）を推定し、より効果的であると思われる箇所から順に教示を促す機能を実装した。

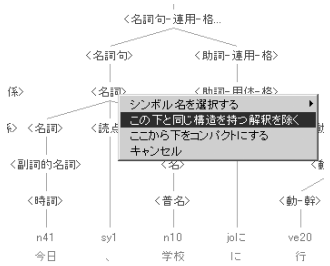

図 8: 間違い箇所を直接教示する

3.1 品詞の教示順序の提示

品詞の教示について、ある形態素の品詞を教示すると曖昧性の減少にどのくらい効果があるかは、その形態素自身が持つ傾向に依ると考えた。そこで、“形態素 m の曖昧度 $A(m)$” を式 (1) のように定義した。

$$A(m) = \frac{1}{N}\sum_{N}\frac{\text{形態素 } m \text{ に品詞を教示する前の構文木数}}{\text{形態素 } m \text{ に品詞を教示した後の構文木数}} \quad (1)$$

式 (1) において分子は、すべての形態素区切りのみを教示した状態の構文木の候補数である。一方、分母は、その状態から形態素 m に対して正しい品詞を教示した後の構文木の候補数である。これらは、本ツールによって正しい構文木を付与した作業結果から求める。$A(m)$ は、両者の比の平均である。ただし、教示前の曖昧性と教示後の曖昧性が変化しなかったもの、つまりその品詞の教示に全く効果がなかったものを除いて平均する。$A(m)$ の計算は形態素区切りのみが教示されていることを仮定しているが、実際に形態素の品詞を教示する場面では、他の形態素や係り受けなどが既に教示されていることも考えられる。しかし本論文では、この差異を無視する。

例えば $A(\text{「ます」}) = 6.2489$ であるとすると、「ます」という形態素に曖昧性が存在する場面でその形態素に対して品詞を教示すれば、曖昧性は 1/6.2489 になると推定される。$A(m)$ の値が高ければ高いほど、その形態素に対して品詞を教示した時に、曖昧性がより減少すると推定できる。したがって、$A(m)$ の値の大きい順に形態素に対して正しい品詞を教示すれば、構文木の数を早期に絞り込むことができる。

本ツールでは、品詞を教示すべき形態素を $A(m)$ の大きいものから順に、図 9 のように提示する。形態素区切りが正しい構文木のみが残されていることを仮定して $A(m)$ の推定を行なっているため、形態素区切りを教示し終わってからこの機能を呼び出すことが望ましい。

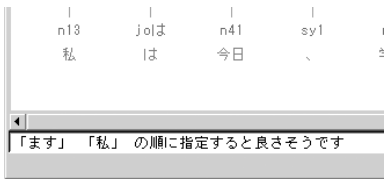

図 9: 品詞を教示すべき順序の提示

3.2 係り受けの教示順序の提示

係り先の教示による曖昧性の減少の効果は、係り元と係り先の文全体に対する位置関係に依存すると考えた。ただし、係り受けを教示される前の段階では、係り先がどこであるかをツールが判断するのは困難であるため、“係り受けの曖昧度 $A(i,l)$” を式 (2) のように定義し、これを係り元の開始位置の文全体の長さに対する割合 i[%] と係り元の長さの文全体の長さに対する割合 l[%] の関数であると考えた。

$$A(i,l) = \frac{\text{該当範囲に係り先を教示する前の構文木数}}{\text{該当範囲に係り先を教示した後の構文木数}} \quad (2)$$

例えば「私は今日、学校に行きます。」という文で係り元が「今日、」である場合、図 10 のように、文全体の文字数を 100 とした時の「私は」の文字数の割合が i であり、「今日、」の文字数の割合が l である。

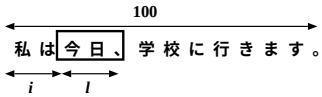

図 10: 文全体の長さに対する割合 i と l

式 (2) において、“係り先を教示する前の構文木数” は、すべての形態素区切りと品詞のみを教示した状態の構文木の候補数である。一方、“係り先を教示した後の構文木数” は、 その状態からその範囲に対して正しい係り先を教示した後の構文木の候補数である。ツールによって正しい構文木を付与した作業結果から i, l, $A(i,l)$ の組を求めて、それらにより関数を最小二乗法で近似した。ただし、教示前の曖昧性と教示後の曖昧性が変化しなかったもの、つまりその係り先の教示に全く効果がなかったものを除いた。$A(i,l)$ の計算は形態素区切りと品詞のみが教示されていることを仮定しているが、実際に係り受けを教示する場面では、他の係り受けなどが既に教示されていることも考えられる。しかし本論文では、この差異を無視する。

本ツールでは、係り先を教示すべき範囲を $A(i,l)$ の値の大きいものから順に、図 11 のように提示する。ただし、ある範囲でまとまることが妥当か否かをツールの側が事前に判断できないため、係り元の範囲として順序付きで提示されるものには適切でないものも含まれ、ユーザは妥当なものを選別する必要がある。

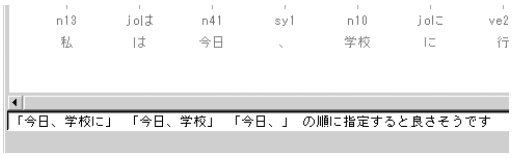

図 11: 係り先を教示すべき順序の提示

3.3 評価実験

ツールによる提示機能がコーパス作成者の負担の軽減にどの程度貢献するかを検証するため、評価を行なった。181 文をテスト文とし、同じ 181 文を推定のための統計データを得るために用いた。この 181 文の平均文長は 38.3 文字、最小文長は 25 文字、最大文長は 59 文字である。また、これらの文を構文解析した結果得られた構文木数の平均は約 8 兆個、最も少ない文で 66 個、最も多い文で約 1300 兆個である。

この機能を使って、品詞に関して曖昧性がある形態素が一つも存在しなくなるまで、ツールが提示した順序に従って正し

表 1: 品詞を教示する順序を提示する機能の評価

	教示した回数	割合
ランダムな順序で教示した場合	828	87.8%
慣れたユーザの直感に基づいた順序で教示した場合	806	85.4%
ツールによって提示された順序で教示した場合	784	83.6%

表 2: 表 1 の内訳

	減少	変化なし	増加
慣れたユーザが自由な順序で教示した場合との比較	18	160	3
ランダムな順序で教示した場合との比較	41	135	5

い品詞を教示し、その回数を調べた。同時に、本ツールの使用に慣れたユーザが自由な順序で品詞を教示した場合と、ランダムな順序で品詞を教示した場合についても調べ、これらを比較した。結果を表 1 に示す。表 1 の最右列に示した数値は、式 (3) によって与えられる評価値である。

$$\frac{\text{教示した回数}}{\text{品詞を教示する可能性がある形態素数}} \times 100 \qquad (3)$$

さらに、構文木を付与する対象となった個々の例文について、ツールによって提示された順序に従って教示した時と、慣れたユーザが自由な順序で教示した時やランダムな順序で教示した時を比較した。結果を表 2 に示す。ツールの提示に従った場合と、慣れたユーザが自由な順序で教示した場合やランダムな順序で教示した場合を比べて、教示する回数が少なくなった文の数を「減少」の欄に示した。同様に、変化しなかった文の数を「変化なし」の欄に、回数が多くなった文の数を「増加」の欄に示した。

表 2 を見ると、どの方法による教示回数の変化がない文が大部分であるが、これは文の解析に用いた文法などの特性のよるものと考えられる。

ツールの提示に従った場合に、ランダムな順序だけでなく、慣れたユーザに比べても、教示回数が少なくなっている。これは、どの形態素の品詞を先に教示すべきか判断することが人間にとって難しく、ツールが人間の苦手な部分をうまく補完していると言える。

係り受けに関しても、同様に評価を行なった。この機能を使って、係り受けに関して曖昧性が一つも存在しなくなるまで、ツールが提示した順序に従って正しい係り受けを教示し、その回数を調べた。同時に、本ツールの使用に慣れたユーザが自由な順序で係り受けを教示した場合と、ランダムな順序で係り受けを教示した場合についても調べ、これらを比較した。結果を表 3 に示す。表 3 の最右列に示した数値は、式 (4) によって与えられる評価値である。

$$\frac{\text{教示した回数}}{\text{係り受けを教示する可能性がある係り元の数}} \times 100 \qquad (4)$$

さらに、構文木を付与する対象となった個々の例文について、ツールによって提示された順序に従って教示した時と、慣れたユーザが自由な順序で教示した時やランダムな順序で教示した時を比較した。結果を表 4 に示す。「減少」と「変化なし」と「増加」の欄の意味は、表 2 と同様である。

表 3: 係り受けを教示する順序を提示する機能の評価

	教示した回数	割合
ランダムな順序で教示した場合	694	50.1%
慣れたユーザの直感に基づいた順序で教示した場合	622	42.9%
ツールによって提示された順序で教示した場合	640	43.9%

表 4: 表 3 の内訳

	減少	変化なし	増加
慣れたユーザが自由な順序で教示した場合との比較	41	87	53
ランダムな順序で教示した場合との比較	82	73	26

表 4 を表 2 と比較すると、品詞を教示する場合と異なり、係り受けを教示する順序によって教示する回数が大きく変化することが分かる。したがって、ツールの側から教示順序を提示する機能の効果を発揮することが、最も期待される。

慣れたユーザに比べてツールの提示に基づく教示回数がわずかながら増えたことから、$A(i,l)$ の推定の精度が不十分であり、係り先の位置関係や係り元の文中での意味的役割など、今回提案した推定手法で用いなかった情報を用いる必要があると考えられる。しかし、ランダムな順序と比較した場合、ツールの教示順序の提示の効果が認められる。したがって、このツールによるコーパス作成に慣れていないユーザでも、慣れたユーザと同じ程度に教示回数を少なくできる。

4. おわりに

4.1 まとめ

構文木付きコーパスを作成するのを容易にするための機能を備えた、構文木付きコーパス作成支援ツールを作成した。このツールによって、構文解析した結果得られる構文木に対して、正しい構文木が満たすべき制約をユーザが教示することによって、正しい構文木を選択することができる。

さらに、品詞と係り受けの教示に関して、どのような順序で教示すると良さそうかということをツールが推定して提示するような機能を実装した。

実装した機能がコーパス作成者の負担の軽減に有効であるかどうかを検証するため、曖昧性がなくなるまでに必要な教示回数で比較しところ、この機能がコーパス作成者の負担を軽減させる効果を持つことが確かめられた。

4.2 今後の課題

本論文で述べたツールに関して、まだ克服すべき課題も多い。

まず、ユーザインターフェイスを改良し、操作方法がユーザにすぐ分かるようにインターフェイスを工夫する必要がある。

また、今回用いた推定方法では、このツールでの作業結果を大量に必要とするという問題点がある。データの蓄積が必要量に比べて圧倒的に不足しているために、出現する形態素すべてに対して品詞の教示についての推定を行なうことができないという理由で、統計データを得るための訓練文をテスト文と同じにして評価を行なった。テスト文以外に大量の作業結果を確保して評価を行なうか、あるいは大量の作業結果を得なくても推定できるように推定方法を改良することによって、この問題点を解決しなければならない。

参考文献

[1] 田中穂積, 亀井真一郎, 森口稔, 加藤安彦. 大きなコーパスを共有しよう. 情報処理, Vol. 41, No. 7, pp. 774–780, 2000.

[2] 白井清昭, 植木正裕, 橋本泰一, 徳永健伸, 田中穂積. 自然言語解析のための mslr パーザ・ツールキット. 自然言語処理, Vol. 7, No. 5, pp. 93–112, 2000.