# 依存構造解析精度向上のためのコーパス整備戦略

大竹 清敬
ATR 音声言語コミュニケーション研究所
kiyonori.ohtake @ atr.jp

## 1 はじめに

依存構造解析は、自然言語処理のために必要とされる重要な処理である。そのため、これまでに、規則に基づく方法、機械学習を用いる方法など様々なアプローチによる解析方法が研究され、解析器が整備されてきた。

しかしながら、現状での解析器の精度は新聞記事を対象とした場合、約 90%程度であり、これをさらに向上させることは困難なように見える。精度を向上させるために、さらなる規則の記述、コーパスの用意などが必要である。一方で、機械学習を前提とする場合、学習に用いる素性を検討することも重要である。ところが、機械学習手法の中には素性の選択に関して頑健で、高い汎化能力を持つものがある。そのような機械学習手法の場合、利用可能な素性をすべて利用しても、性能が大きく低下することはない。実際に、Sasano は SVM を用いる依存構造解析器で利用する素性を拡張した実験を行っているが、その性能差はわずかである [Sas04]。

依存構造解析器のための規則を記述する上でも、また、機械学習手法を用いて解析器を実現するためにも正しく解析されたコーパスを蓄積することは有用である。このような依存構造解析済みのコーパスは、依存構造解析器の実現のみならず、自然言語処理における重要な言語資源としても利用可能である。ところが、依存構造解析済みコーパスの整備は安価に行えず、それを大量に用意することは容易ではない。また、高精度の依存構造解析器を実現するために、どのように依存構造解析済みコーパスを整備していったらよいかの具体的な指針や戦略がはっきりしていない。

そこで、本稿では、機械学習に基づく依存構造解析器を前提として、解析精度を向上させるためにどのようにコーパスを整備するべきかについて議論する。

## 2 依存構造解析済みコーパス

本稿では、依存構造解析済みコーパスとして京都テキストコーパスならびに IPAL 各辞書に含まれる例文を依存構造解析し、修正したコーパスを用いる。

### 2.1 京都テキストコーパス

京都テキストコーパス（Version 3.0）は、1995 年の毎日新聞記事のうち 1 月 1 日から 1 月 17 日の一般記事（約 2 万文）ならびに、1999 年のすべての社説記事（約 2 万文）から構成される。これらの記事を JUMAN と KNP によって解析し、人手によって修正したものである。また、現在は、さらにこのうちの 5,000 文に対して格関係、照応・省略関係、共参照の情報を付与した Version 4.0 が公開されている[1]。本研究で使用したのは、京都テキストコーパス（Version 3.0）を形態素解析器 chasen が使用する品詞体系へ変換したものである。変換後、文節区切り基準の修正などを施した。京都テキストコーパスの諸表を表 1 にまとめる。なお、一般記事と社説をあわせた場合の語彙サイズ[2]は、34694 である。

表 1: 依存構造解析済みコーパス

| | 京都テキストコーパス（一般記事） | 京都テキストコーパス（社説） | IPAL コーパス |
|---|---|---|---|
| 文数 | 19669 | 18714 | 15330 |
| 文節数 | 192154 | 171461 | 67170 |
| 形態素数 | 542334 | 480005 | 156131 |
| 語彙サイズ | 29542 | 17730 | 11895 |
| 文節/文 | 9.769 | 9.162 | 4.382 |

### 2.2 IPAL コーパス

本研究では、計算機用日本語名詞辞書 IPAL、計算機用日本語基本動詞辞書 IPAL、計算機用日本語基本形容詞辞書 IPAL の 3 辞書に含まれる例文（それぞれ 7720 文、5244 文、2366 文）を依存構造解析器 CaboCha[3] によって解析し、人手で整備した。以下、このコーパスを IPAL コーパスと呼ぶ。IPAL コーパスの諸表を表 1 に示す。

## 3 実験

### 3.1 解析誤り解析

まず、基本的な依存構造解析実験として CaboCha を用いて交差検定を行った。交差検定を行うために各コーパスを任意に 2 分割した。2 分割した結果をそれぞれ京都テキストコーパスの一般記事は IP0, IP1、社説は ED0, ED1、IPAL コーパスは IPAL0, IPAL1 として表記する。実験結果を表 2 に示す。表における「学習」と「テスト」はそれぞれ、学習とテストに用いたコーパス

[1] http://www.kc.t.u-tokyo.ac.jp/nl-resource/corpus.html
[2] 用語として語彙サイズを用いるが実質的には形態素のことなり数である。
[3] http://chasen.org/~taku/software/cabocha/

を示している。SVM の核関数には多項式関数を用いているが、その次数は 2 または 3 であり、それを「次数」にて示す。「精度」は係り受けの正解率であり、「文正解率」は完全に正しく解析された文の割合を示している。なお、文節まとめ上げまでを行ったデータを入力とし実験を行った。したがって、実際の状況では、ここで示した性能が、形態素解析、文節まとめ上げの性能によってさらに低下すると考える。

表 2: 交差検定結果

| 学習 | テスト | 次数 | 精度 (%) | 文正解率 (%) |
|---|---|---|---|---|
| IP0 | IP0 | 2 | 94.06 | 65.51 |
| IP0 | IP0 | 3 | 99.96 | 99.71 |
| IP0 | IP1 | 2 | 89.50 | 50.35 |
| IP0 | IP1 | 3 | 89.23 | 49.33 |
| IP1 | IP0 | 2 | 89.60 | 49.89 |
| IP1 | IP0 | 3 | 89.21 | 49.05 |
| ED0 | ED1 | 2 | 90.77 | 55.58 |
| ED1 | ED0 | 2 | 90.52 | 54.62 |
| IPAL0 | IPAL1 | 2 | 97.43 | 92.25 |
| IPAL1 | IPAL0 | 2 | 97.69 | 93.06 |
| IP0 | IPAL0 | 2 | 97.76 | 93.15 |
| ED0 | IPAL0 | 2 | 97.56 | 92.81 |

実験結果の中から京都テキストコーパスの一般記事 (IP0 と IP1) における交差検定の結果のうち多項式関数の次数が 2 のものについて誤りを分析した。この交差検定における平均の解析精度は 89.55 (154455/172485)%、文正解率は 50.12 (9858/19669)% である。

分析したのは、誤りにおける係り元の品詞系列とその文字列、誤って係ると判断した文節までの距離、また正解とされる文節までの距離を調べた。まず、誤り係り元の品詞系列の上位 10 件を表 3 に示す。次に、同じく誤り係り元の文字列とその頻度を調べた結果の上位 10 件を表 4 に示す。

表 3: 誤りの係り元品詞系列とその頻度（上位 10 件）

| 品詞系列 | 頻度 |
|---|---|
| 名詞-一般, 助詞-格助詞-一般 | 835 |
| 動詞-自立, 記号-読点 | 576 |
| 名詞-一般, 助詞-係助詞 | 444 |
| 名詞-副詞可能, 記号-読点 | 370 |
| 動詞-自立 | 336 |
| 名詞-数, 名詞-接尾-助数詞, 記号-読点 | 318 |
| 名詞-一般, 助詞-連体化 | 304 |
| 副詞-一般 | 304 |
| 動詞-自立, 助動詞 | 281 |
| 動詞-自立, 助詞-接続助詞, 記号-読点 | 265 |

次に、誤った係り受けにおける係り先までの距離（誤り距離）とその時の正解までの距離（正解距離）の関係について調べた。調査結果のうちその頻度が大きいもの上位 10 件を表 5 に示す。表中の誤りとは誤った係り先までの距離を意味し、1 の場合は直後の文節となる。正解は、本来かけるべき文節（正解文節）までの距離を示している。

表 4: 誤り係り元の文字列とその頻度（上位 10 件）

| 文字列 | 頻度 | 文字列 | 頻度 |
|---|---|---|---|
| ため、 | 76 | ある | 30 |
| うち | 41 | あり、 | 29 |
| さらに | 39 | うち、 | 25 |
| ため | 34 | なり、 | 25 |
| ほか、 | 33 | もう | 24 |

表 5: 誤り距離と正解距離とその頻度

| 誤り | 正解 | 頻度 | 誤り | 正解 | 頻度 |
|---|---|---|---|---|---|
| 1 | 2 | 3117 | 2 | 4 | 478 |
| 2 | 1 | 1362 | 3 | 2 | 436 |
| 3 | 1 | 919 | 4 | 1 | 434 |
| 1 | 3 | 863 | 4 | 2 | 379 |
| 2 | 3 | 482 | 1 | 4 | 329 |

### 3.2 想定実験

この節では、実際にコーパスを拡充することを想定した実験を行う。具体的には、京都テキストコーパスの一般記事である IP1 が与えられた状況で社説記事のコーパスを拡充する状況を考える。この時、可能な戦略として、(1) 長い文を優先、(2) 語彙を豊富にする文を優先、(3) 記事が書かれた時間順、(4) ランダムという 4 つの選択基準を考慮し、同程度の大きさコーパスをそれぞれ IP1 に加え、これらを学習コーパスとして CaboCha で学習し評価する。

学習コーパスとして次の 5 種類を用意した。(1) IP1+LONG、(2a) IP1+VsortReg、(2b) IP1+Vsort、(3) IP1+Chrono、(4) IP1+Rand(ED0) である。ここで、(1) の LONG は文の長さを基準として降べきに並び替えたコーパスである。文の長さは、文字列長ではなく、文節数を基準とした。(2a) と (2b) の違いはいずれも形態素を単位とする語彙を最大化するように並び替えたコーパスを IP1 に付加したものであるが、並び替えの基準が異なる。いずれも語彙サイズ 0 から開始して語彙サイズを大きくする文を選択していくことによって並び替える。VsortReg は語彙をより大きくする文を選択する際に、増加する語彙数をその文の形態素数で割った値が大きいものを選択していく。したがって最初の文を選択する時、この値はすべての文で 1 となる。実際にソートするときは最初の文を任意に決める。一方、Vsort は単純に語彙数を最大化する基準で文を選択する。したがって、Vsort で一番最初に選ばれる文は語彙を最大にする非常に長い文となる。(3) は記事が書かれた時間順に IP1 に追加したもの、(4) は任意に選択した文（実際には ED0）を IP1 に追加したものである。

これら 5 つの基準で京都テキストコーパスの社説を並び替えた場合の語彙サイズの変化を図 1 に示す。縦軸に語彙サイズ、横軸に文数をとり、500 文毎の語彙サイズを示している。ここに表示したのは京都テキストコーパスの社説をすべて対象としている。実験では、追加するコーパスの量の基準として係り受け数を考え、並び替えたそれぞれのコーパスの先頭から同程度の係り受け数となるように文を選択し、追加する各コーパスを構成した。

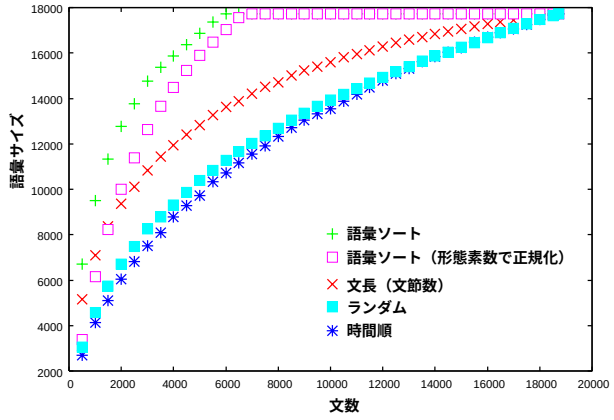


図 1: 並び替えたコーパスとその語彙サイズ

各コーパスの基本情報を表 6 にまとめる。

表 6: 各コーパスの基本情報

| 名前 | 文数 | 文節数 | 語彙 | 係受数 | 文節/文 |
|---|---|---|---|---|---|
| IP1 | 9834 | 95871 | 21572 | 86037 | 9.749 |
| LONG | 5490 | 81759 | 13266 | 76269 | 14.89 |
| VsortReg | 9072 | 85345 | 17730 | 76273 | 9.408 |
| Vsort | 8348 | 84618 | 17730 | 76270 | 10.14 |
| Chrono | 9342 | 85609 | 13278 | 76267 | 9.164 |
| Rand | 9357 | 85628 | 13561 | 76271 | 9.151 |
| K-mag | 489 | 4851 | 2501 | 4362 | 9.920 |
| IPAL0 | 7665 | 33484 | 8617 | 25819 | 4.368 |
| IP0 | 9835 | 96283 | 21616 | 86448 | 9.790 |

評価のために、異なるドメインでなおかつ、豊富な話題を含むものとして小泉内閣メールマガジン創刊準備号（2001 年 5 月 29 日）から 7 月 19 日発行分までの 7 号分のテキストとその正解依存構造データ (K-mag とする) を用意した。さらに IPAL コーパスを任意に 2 分割したうちの IPAL0 ならびに同一ドメインのコーパスとして IP0 をテストセットとして用いた。

実験の条件は、文節まとめあげまでを行ったデータを入力とした。また、SVM の核関数は 2 次の多項式関数である。

K-mag を解析した結果を表 7 に、IPAL0 を解析した結果を表 8 に、IP0 を解析した結果を表 9 にそれぞれ示す。

表 7: K-mag の解析結果

| 学習コーパス | 精度 (%) | 文正解率 (%) |
|---|---|---|
| IP1 | 87.25 | 49.69 |
| IP1+LONG | 87.67 | 51.53 |
| IP1+VsortReg | 87.55 | 50.92 |
| IP1+Vsort | 87.53 | 50.31 |
| IP1+Chrono | 87.57 | 50.31 |
| IP1+Rand | 87.60 | 49.69 |

表 8: IPAL0 の解析結果

| 学習コーパス | 精度 (%) | 文正解率 (%) |
|---|---|---|
| IP1 | 97.68 | 93.02 |
| IP1+LONG | 97.75 | 93.22 |
| IP1+VsortReg | 97.77 | 93.28 |
| IP1+Vsort | 97.73 | 93.12 |
| IP1+Chrono | 97.71 | 93.06 |
| IP1+Rand | 97.69 | 93.06 |

# 4 考察

## 4.1 解析誤り解析

機械学習手法に基づく依存構造解析器として SVM を用いる CaboCha を利用した。一般に核関数を用いて識別問題を解こうとするとき、問題に対する最適な核関数をあらかじめ知ることは難しい。これまで、依存構造解析に対しては 2 次あるいは 3 次の多項式関数が経験的に用いられてきた。実験では、3 次の多項式関数を用いた場合に、学習データに対する再現性（クローズドテスト）は非常に高い（ほぼ 100%の）性能を示すが、交差検定（オープンテスト）を行った場合に、2 次の多項式関数ほど高い精度を示せなかった。これは、3 次の多項式関数を利用した場合にオーバーフィットしているともいえるが、利用状況によっては、3 次の多項式関数を用いた方が好ましい場合も考えられる。

一方、2 次の多項式関数を用いてクローズドテストを行った場合、精度は、94%ほどであり、2 次の多項式関数を用いる場合の上限であると考える。この値は、SVM のソフトマージンパラメータを変更しても大きな変化は見られなかった。しかしながら、学習コーパス中に不整合性があり、うまく再現できていない箇所も存在するため、学習コーパスをさらにクリーニングすることによって改善が見込める。

また、表 2 の最下部に IPAL0 を IP0 と ED0 でそれぞれ解析した結果を掲載した。この結果から、1 文あたりの文節数が学習に用いたコーパスの半分以下であるコーパスを解析する場合、ドメインの違いはほとんど問題にならず、解析対象の文が短かいため約 97%で解析できることがわかる。

次に実際の解析誤りの性質について目を向ける。表 3 に示した結果から、名詞と格助詞の組み合わせである格要素が、どの動詞に支配されるのかの判断を誤っている事が多いと推測できる。一方で、2 番目には動詞の係り先を誤る事例があり、並列構造の同定が難し

表 9: IP0 の解析結果

| 学習コーパス | 精度 (%) | 文正解率 (%) |
|---|---|---|
| IP1 | 89.60 | 49.89 |
| IP1+LONG | 89.99 | 51.25 |
| IP1+VsortReg | 89.97 | 51.31 |
| IP1+Vsort | 89.90 | 51.22 |
| IP1+Chrono | 89.86 | 51.09 |
| IP1+Rand | 89.95 | 51.20 |

い事であることを示している。3 番目には名詞と係助詞の組み合わせが位置している。これは、主題提示の機能をもち、京都テキストコーパスの基準では、かかりうる述部のうち最後尾にかける仕様であるため長距離依存関係になりやすく、誤りやすい。

係り先を誤りやすい具体的な文字列として、「ため」や「うち」など節の終端にくるものが多いことが表 4 からわかる。これは、文が長くなり、複数の節から構成される場合に、節間の関係を適切に捉えることが難しいためと考える。

さらに、表 5 から機械学習の結果は、直後の文節にかける傾向があるが、実際の正解は 2 文節先の場合が非常に多いことが分かる。これは「A の B の C」や「[連体節]A の B」といった頻出するパターンが該当する。また、「[連体節]A の B」というパターンの CaboCha による係り受け精度が 59.6%であるという報告 [Tak04] があり、解析が困難なパターンであると考える。一方で、「パソコンの/処理の/速度」のように判断が難しいものも存在し、わずかながらコーパス中で整合性がとれていない箇所も存在すると考える。

### 4.2 想定実験

想定実験を通して、コーパスを整備する上での可能な戦略をいくつか試した。しかし、実験の規模が小さいためか、得られた結果には、大きな違いは見られなかった。

全体の傾向として、解析精度は長距離依存を含むような長い文がどの程度あるかに支配される。単純に京都テキストコーパスを一般記事と社説に分けた上で交差検定した場合でも、1 文あたりの平均文節数が大きい一般記事 (9.769) の方が社説 (9.162) に比べて精度が約 1%ほど劣る（表 2)。この原因は長距離依存の係り受けを誤った場合に、非公差条件の制約などから他の誤りを誘発するからである。したがって、解析結果の傾向として文が長ければ長いほど解析誤りがそこに集中する可能性が高まる。このことから、解析精度を向上させるためには、いかに長距離依存の係り受けを正しく解析するかが重要となる。実際に、長い文を含むテストセットを対象とした実験、たとえば K-mag や IP0 を対象とした実験では、長い文を優先的に追加した方がより高い性能を示した。

しかしながら、一般に長い文を整備するすることは、人間にとっても負担が大きい。同程度の係り受け数を持つコーパスを整備する事を考えると、1 文あたりの文節数が小さい方が総作業時間も短くなると予想する。実際問題として長文の解析精度を向上させるためには、文を分割し処理する枠組みを考えることが有望である。たとえば、文を節単位へ分割することはある程度高精度にできることが分かっているので [Mar04]、CBAP などを用いて節分割を行い、節内の最終文節を除く文節が他の節内の成分へ係るような明らかな誤りは抑制できる。しかしながら、この場合でも、節間の依存関係をどのように明らかにするかという問題は、残されており、現状の依存構造解析器にとっても解析が難しい。

コーパスを整備していくときの戦略の一つとして語彙サイズに着目した。しかし、精度を上げるためには、長い文を優先的に追加した方が有利なこともあり、その効果を示せなかったと考える。語彙的に豊富なコーパスの方が、未知となる形態素が減少することから近距離依存における誤りも減少すると予想した。ところが、依存構造の判定には最低でも係りと受けの 2 文節に関する情報が必要である。実際にはその周辺の情報も利用するため、未知となる依存構造は組み合わせ的に爆発する。そのため、表 8 に示された結果でも、語彙サイズを大きくする文から優先的に追加した場合の方が、長さを優先させる場合よりも数値上すぐれている。しかし、その違いはほんのわずかである。また、語彙サイズは大きくとも、品詞の情報だけで十分に精度良く判定できる現象も多くあり、性能向上に寄与しないデータも数多く存在すると考える。

## 5 むすび

本稿では、機械学習を前提とした依存構造解析器の精度を向上させるためのコーパス整備戦略について議論した。実験では、長い文を優先的に学習コーパスへ追加した場合が高い精度を示した。これは、長距離依存の関係を誤ることによって連鎖的に他の誤りを引き起こす場合があるためである。一方で、本稿にて行った実験のために、依存構造解析済みコーパスとして既存の京都テキストコーパスの他に、IPAL の各種辞書の例文を解析したコーパス、ならびに小泉内閣メールマガジンの一部の解析済みコーパスを整備した。

本研究は総務省の研究委託により実施したものである。

## 参考文献

[Mar04] 丸山岳彦, 柏岡秀紀, 熊野正, 田中英輝：日本語節境界検出プログラム CBAP の開発と評価, 自然言語処理, Vol. 11, No. 3, pp. 39–68 (2004).

[Sas04] Sasano, M.: Linear-Time Dependency Analysis for Japanese, In *Proceedings of Coling 2004*, pp. 8–14 (2004).

[Tak04] 竹伸伸介, 徳久雅人, 村上仁一, 池原悟：動詞節に修飾された名詞句の係り受け解析, 情報処理学会 研究報告 NL-163, pp. 83–89 (2004).