品詞タグ付きコーパスを直接利用した形態素解析

山下達雄，松本裕治
{tatuo-y,matsu}@is.aist-nara.ac.jp
奈良先端科学技術大学院大学 情報科学研究科

1．はじめに

最近の形態素解析システムは、 コーパスから統計的に語の出現確率や連接確率を推定し それを用いるという手法が一般的である。 しかし、 このような確率によるデータの抽象化手法には、 例外的な現象に対応できないなどの弊害がある。 これに対処するために、近年、 今まで切捨ててきた言語現象を連語登録[1]や可変長規則[2]といった 実際のデータに近い形で補助的に利用する研究が行なわれている。

一方、実際のデータを補助的に用いるのであれば、 始めからデータを加工せずにそのまま使用して解析を行なうという方法が考えられる。 この方法は、記憶装置、演算装置、高速文字列検索[3]などの技術の進歩を考えれば、 現在の段階で十分実用可能であると言える。 本研究では、この考えに基づき、 品詞タグ付きコーパスをそのままデータとして用いる形態素解析手法を提案する。

2．処理の概要

本研究で提案する手法では、 まず、解析したい文の部分文字列をキーとして品詞タグ付きコーパスを検索し、 見つかったコーパス片に付与されているタグをそのまま利用することで、利用形態素解析を行なう。 簡単な例を用いて処理の概念を説明する。 以下のような三文からなる品詞タグ付きコーパスがあるとする。

(1) 昨日[名詞] 、[記号] 太郎[名詞] が[助詞] 走った[動詞] 。[記号]
(2) 花子[名詞] は[助詞] ラーメン[名詞] を[助詞] 食べた[動詞] 。[記号]
(3) 彼[名詞] は[助詞] 走った[動詞] 。[記号]

解析したい文「昨日、花子は走った。」に対して、 「昨日、」という部分は(1)の「昨日[名詞] 、[記号]」、 「花子は」という部分は(2)の「花子[名詞] は[助詞]」、 「は走った。」という部分は(3)の「は[助詞] 走った[動詞] 。[記号]」を 利用してタグ付けを行なえば、 以下のような形態素解析結果を得ることができる。

<table>
<tr><td>文</td><td>《</td><td>昨</td><td>日</td><td>、</td><td>花</td><td>子</td><td>は</td><td>走</td><td>っ</td><td>た</td><td>。</td><td>》</td></tr>
<tr><td rowspan="3">コーパス片</td><td></td><td colspan="2">名詞</td><td>記号</td><td colspan="2"></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td colspan="2">名詞</td><td>助詞</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>助詞</td><td colspan="3">動詞</td><td>記号</td><td></td></tr>
<tr><td>結果</td><td></td><td colspan="2">名詞</td><td>記号</td><td colspan="2">名詞</td><td>助詞</td><td colspan="3">動詞</td><td>記号</td><td></td></tr>
</table>

実際には、コーパス片の重なり部分に形態素の曖昧性が生じることがあるので、曖昧性解消の基準を設けておく必要がある。 これについては後で詳しく述べる。
ここで、この手法の利点をいくつかあげてみる。

- 頻度の低い言語現象も扱うことができる。
- 解析失敗の原因(解析の基になるコーパスのタグ付け間違い)がすぐに分かるので品詞タグ付けコーパスの保守作業に有用。
- 解析結果をコーパスに追加すればすぐに反映される(現在の検索用データ構造は Suffix Array なので不可能だが、木構造を採用すれば可能になる)。
- 同じ文がコーパスにあれば解析精度はほぼ100%になる。
- 頑健な解析が可能。スペルチェッカーなどに応用可能。

また、 自然言語処理を応用した高度な検索のために、今後、 意味や構造などのタグのついた日本語テキストが蓄積されていくと期待できる[4]。 この手法では、そのようなデータを検索だけでなく解析にもそのまま利用できる。 例えば、タグ付きデータを大量に保有しているサイトでは、 そのデータに対しての検索質問文などの解析を効率的に行なうことができる。

このようなタグ付きデータをそのまま利用した方法として ATR の νｙ(NUU)-Talk 音声合成システム[5]が知られている。 νｙ-Talk では、入力音声記号列に対して、 蓄えられたタグ付き音声データからできるだけ使用環境に近いスペクトルを選ぶという方針により音声合成を行なっている。

3．システムの詳細

本研究では、 システム構築にあたって以下の目標を設けた。

- 言語知識を用いない： 特定の言語に依存せず、あらゆるタグ付きデータに利用できる汎用システムを目指す。
- 統計情報を用いない： 統計情報などの外部データがなくとも動く身軽なシステムを目指す。

とりあえず、汎用システムの構築を最優先とし、 形態素解析精度を上げるためのテクニック(言語知識・統計情報の付加など)は今回は用いない。
システム構築には、プログラング言語Perlを主に用い、 検索部分はC(SUFARY[3]パッケージ)を使用した。

3.1．高速全文検索のための品詞タグ付きコーパスのフォーマット

2章で説明したように、本手法では、 まず、品詞タグ付きコーパスを検索して品詞列を得る処理を行なう。 これを高速で行なうために、 各文字に対して品詞を表すコードを割り当てる方法を用いた。
一般に品詞タグ付きコーパスは2章の例で示したように形態素ごとに分割されているフォーマットが多い。 このようなフォーマットでは形態素をまたぐ文字列の検索には複雑な文字列処理が必要となる。 そこで、以下のように、 品詞タグ付きコーパスを文とタグに分けて、文字位置で対応が取れるようなフォーマットを利用することにした。

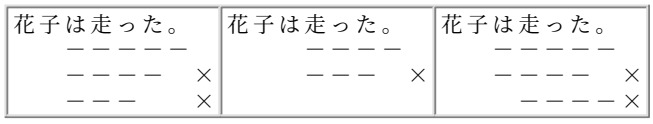
(1) 花子は走った。 (M)M|J(V-V)V|T
(2) 彼は食べた。 M|J(V-V)V|T

各文字に対してその文字が含まれる形態素の品詞とその文字の形態素内での位置を表すタグが付与される。 以下に「花子は走った。」という文とそれに付与されているタグの意味を示す。

位置	0	2	4	6	8	10	11
文	花	子	は	走	っ	た	。
タグ	(M	)M	|J	(V	-V	)V	|T
意味	「名詞」を構成する最初の文字	「名詞」を構成する最後の文字	「助詞」を構成する唯一の文字	「動詞」を構成する最初の文字	「動詞」を構成する途中の文字	「動詞」を構成する最後の文字	「記号」を構成する唯一の文字

このフォーマットにより部分文字列を高速に検索することができ、その文字列に対応する品詞タグも素早く容易に取り出すことができる。 例えば、 図の例(「花子は走った。 (M)M|J(V-V)V|T」)を品詞タグ付きコーパスとみなし、 「は走った」という文字列にどのような品詞が振られている例があるのか検索することを考える。 「は走った」は「花子は走った。」という文字列の4文字目から12文字目なので、 品詞タグ列「(M)M|J(V-V)V|T」の4文字目から12文字目の文字列「|J(V-V)V」が、対応する。 よって、「は」は助詞、「走った」は動詞ということが容易に分かる。

3.2．解析対象文の部分文字列の検索



ここでは実際に品詞タグ付きコーパスを検索する方法について説明する。 検索部には高速全文検索を行なうシステムSUFARY[3]を用いている。
検索は文頭から文末に向かって行なわれ、各位置から品詞タグ付きコーパスとの最長一致部分文字列だけが、 検索結果として後の処理に利用される。 ただし、他の最長一致部分文字列に含まれるような文字列は削除される。 右に検索の例を示す(×は削除された検索結果を表す)。
今回は、検索結果に部分的にしか一致しない形態素が混じっていた場合は無視することにする。例えば、検索結果「は走っ」では「走っ」が「走った[動詞]」の一部でしかなく、 半端な検索結果として削除される。

3.3．最適解の選択

前節の処理により選ばれた検索結果(コーパス片と呼ぶ)に含まれる形態素列を基に形態素解析結果としての解を選択する処理について説明する。 選択のための優先条件として以下のものを考慮した。

1．他の形態素と連接しない形態素は優先度が低い。
2．コーパス片が重なっている部分の共通の形態素は優先度が高い。

3．長いコーパス片に属する形態素は優先度が高い。
4．長い形態素の優先度は高い。

これらをコストなどの体系にまとめるのは困難なので、 優先条件の適用の順序を決めてそれに従って選択処理を行なうことにした。

他の形態素と連接しない形態素の削除

直前・直後に接続できる形態素がなく、かつ、他の対抗する形態素よりも文字列長が短い形態素を浮き形態素と呼ぶことにする。 最適解に浮き形態素が選択されると未定義語を出力する必要がでてきてしまうので、 検索結果のコーパス片に含まれている浮き形態素を全て削除することにする。 例えば、右図では「です[助動]」が浮き形態素である。 これを最適解に選択すると、「む」を未定義語にしなければならない。 それよりも素直に「で[助詞] すむ[動詞]」を選択した方が現実的である。 結局、浮き形態素は最適解としてふさわしくないと思われるので、 早い段階で削除する。

文	《	で	も	、	し	な	い	で	す	む	》
コーパス片		接続		記号							
		助詞	助詞	記号							
					動詞	助動		助詞			
						助動		助動			
								助動	動詞		

曖昧性の解消

曖昧性の解消は、 検索結果に含まれる全ての形態素の共通の区切りに着目し、 それらで区切られた部分ごとに行なう。 上の例では、{でも，、，し，ない，で，すむ} の7つの部分に分割される(「です[助動]」は浮き形態素として削除されてるので考慮しない)。 共通の区切りによって分割された部分を共通区切り領域と呼ぶことにする。 処理の手順としては、 まず検索結果を共通区切り領域に分割し、それから各領域ごとに最適形態素の選択を行なう。
共通区切り領域は以下の3つのタイプに分類できる。

Type 1	Type 2	Type 3
共通区切り領域内の全ての形態素の長さとタグがまったく同じである場合。	共通区切り領域内の形態素の長さは全て同じだがタグが異なるものがある場合。	共通区切り領域内の形態素に長さが異なっているものがある場合。
…山下… …名詞 名詞…	…これ… …動詞 名詞…	…で は… …助詞 助詞 接続…

Type 1 の場合は、曖昧性がないので、そのままそのタグが最適解として決定される。 Type 2，3 の場合は、 今回は以下の手順に従って曖昧性の解消を行なった。

1	共通の形態素があればそれを選択する。	…… 名詞 …名詞… 動詞…… 名詞
2	それで決まらなければ、各形態素の属するコーパス片の文字列長を比較し、 それが最長のコーパス片に属する形態素を選択する。	…… 助詞 動詞… 助詞
3	それでも決まらなければ、各形態素の長さを比較し、最長の形態素を選択する。	…助詞 助詞… … 接続詞 … 接続詞
4	それでも決まらなければ、適当に選ぶ。	

4．実験と考察

本手法と統計的手法とを比較するための実験を行なった。 品詞タグ付きコーパスをそのまま用いたシステム(BTG)と、 そのコーパスから統計的にパラメータ(bigram)を学習しそれを用いた形態素解析システム茶筌[6]とで、アウトサイドデータを解析し精度を測定した。
実験には毎日新聞95年版の品詞タグ(IPA品詞体系)付きコーパス[7]を使用した。 １万文(約26万形態素)、２万文(約52万形態素)、３万文(約78万形態素) を学習/検索用コーパスとし、 これらとは別の2000文(形態素数 約5万3千)を評価用コーパスとした。 茶筌の辞書項目は学習コーパスから抽出した形態素のみを使用している。 解析結果を評価用コーパスと形態素ごとに比較して、 品詞の一番荒

い分類(名詞、動詞、助詞といったレベル)が異なるものを誤りとみなした。 評価結果を表に示す。

	茶筌 適合率/再現率	BTG 適合率/再現率
１万文	87.1% / 92.3%	88.1% / 88.9%
２万文	88.7% / 92.8%	90.4% / 91.8%
３万文	90.1% / 93.5%	91.7% / 93.2%

両手法の解析結果を比較すると、誤りを生じる場所に違いが多い。 解析精度はほぼ同じあることから、 これは、それぞれの手法の特徴が解析に反映されているということを意味する。

本手法の欠点として、 活用などの言語情報を利用しないので、活用語の解析精度が落ちるという点があげられる。 茶筌ではプログラム内部で活用処理をしているので、この部分の精度の差は歴然としている。 これは品詞体系との相性(活用語をどう扱うか)の問題といえる。 また、 本手法では、検索用コーパスの量が少ないと、短いコーパス片に現れる形態素を採用してしまうことが多い。 これにより本手法の特徴である文脈(前後の文字列)が考慮できず当然精度は落ちる。 優先条件が活かされず、曖昧性の解消も不正確になる。これはスパースネスの問題である。

一方、本手法の特徴が活かされている部分も少なくない。 例えば、「障害はなくなりつつある。」 という文を茶筌で解析した結果は以下のようになる。

障害[名詞] は[助詞] なく[助動詞] なり[動詞] つつ[助詞] ある[動詞] 。[記号]

評価用コーパスでは「なくなり[動詞]」が正解とされているので、 「なく[助動詞] なり[動詞]」の部分は誤りである。 この誤りは、助動詞「ない」と動詞「なる」の出現確率(つまり出現頻度)が 動詞「なくなる」に比べ非常に高いことが原因であった。 一方、BTGでは同じ文を誤らずに解析できる。 これは、検索用コーパスに、

今[名詞] は[助詞] なく[助動詞] なり[動詞] つつ[助詞] ある[動詞] 。[記号]

という文が存在し、「はなくなりつつある。」という部分がそのまま流用できるからである。

これらのことにより、 本手法で十分な精度が得られない部分に、 統計情報などを用いた従来の形態素解析手法を補助的に用いることにより、 より高い解析精度を得ることが可能だと思われる。 つまり、実際のデータに答えがあるのならばそのまま利用してしまえばいいのであり、 それでうまくいかない部分は従来の手法を利用するということである。

また、現在入手できる日本語品詞タグ付きコーパスには、 一貫性の欠如に起因する誤りが多い。例えば、同じ「大阪駅」という文字列に対する品詞タグが 「大阪[名詞] 駅[名詞]」であったり「大阪駅[名詞]」であったりする。 本手法は、 品詞タグ付きコーパスの全文検索という処理を行なうので、 この種のタグ付け誤りを効率的に発見することができ、 品詞タグ付けコーパス修正支援ツールとしての用途が期待できる。

5．おわりに

今後の展開として、 品詞タグ付きコーパスを直接利用する手法の 統計的手法との統合による解析精度の向上、および、 品詞タグ付けコーパス修正支援ツールとしての利用 を目指し研究を進めたい。 また、形態素解析以外の自然言語処理への適用も考えていきたい。

追記： 本研究に関しての詳細情報は http://cactus.aist-nara.ac.jp/~tatuo-y/doc/tg/ を御参照下さい。

参考文献

1. 山地治、黒橋禎夫、長尾眞: "連語登録による形態素解析システムJUMANの精度向上" 言語処理学会第2回年次大会発表論文集, pp.73-76, March 1996.
2. 北内啓、山下達雄、松本裕治: "日本語形態素解析システムへの可変長連接規則の実装", 言語処理学会第3回年次大会, pp.437-440, March 1997.
3. "Suffix Array を用いた高速文字列検索システム SUFARY", http://cactus.aist-nara.ac.jp/lab/nlt/ss/
4. 橋田浩一、杉村領一、柏岡秀紀、内山将夫、Christoph J.Neumann: "大域文章修飾: 標準タグによる言語データの大規模な構造化と再利用", 言語処理学会第3回年次大会, pp.135-138, March 1997.
5. ATR国際電気通信基礎技術研究所 編: "ATR先端テクノロジーシリーズ 自動翻訳電話", オーム社, 1994.
6. "形態素解析システム『茶筌』", http://cactus.aist-nara.ac.jp/lab/nlt/chasen.html
7. "RWCテキストデータベース報告書", 新情報処理開発機構(RWCP), 1997.