社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
TL99-44 (2000-03)

実例からの帰納的学習を用いた構文解析手法

正富 欣之　荒木 健治　栃内 香次

北海道大学大学院工学研究科
〒060-8628 札幌市北区北13条西8丁目
TEL: 011-706-6823　FAX: 011-709-6277
E-mail: {tome,araki,tochinai}@media.eng.hokudai.ac.jp

あらまし　自然言語処理における構文解析手法に、固定された文法規則を用いるものと用例から得られたデータを用いるものがある。これらの手法における問題点として、前者は対象に依存した言語現象への対処が困難である、後者はデータ量が膨大になる、ということが挙げられる。そこで、実例から帰納的学習を用いて構文解析規則を獲得し、得られた解析規則を適用した構文解析を行うことにより、上記の問題を解消することを目指した手法を提案する。本手法の利点としては、学習アルゴリズムは言語に依存していないので、学習データさえあれば同一のアルゴリズム（帰納的学習）でどのような言語にも適用できるということがある。すなわち，多言語対応構文解析手法としての能力を有するものと考えられる，本稿では、上述の手法に基づいたシステムについての概要を述べる，また、日本語文5,000文を用いた実験を行い、その結果について考察し、本手法の有効性について検討する。

キーワード　構文解析、帰納的学習、多言語

A Syntactic Analysis Method Using Inductive Learning from Examples

Yoshiyuki MASATOMI, Kenji ARAKI and Koji TOCHINAI

Graduate School of Engineering, Hokkaido University
N13 W8, Kita-ku, Sapporo, 060-8628 Japan
TEL: +81-011-706-6823　FAX: +81-011-709-6277
E-mail: {tome,araki,tochinai}@media.eng.hokudai.ac.jp

Abstract　In this paper, we propose a syntactic analysis method using inductive learning from examples. In our proposed method, the system acquires the parsing rules using the examples of parsing result. And the system parses Japanese sentences using the acquired parsing rules. We consider that our proposed method can resolve problems of the Rule-based approach and the Example-based approach. Moreover, we consider that this method can be applied to the other languages. We performed the experiment using EDR Japanese Corpus. We consider the result of experiment about the ability of our proposed method.

key words　parser, inductive learning, multi-language

1 はじめに

自然言語処理において，構文解析は文を解析する上で非常に重要である．それゆえ，これまで構文解析に関する多くの研究がなされてきた．それらの研究で，主となる構文解析手法は構文解析規則を用いたものである[1][2]．これらの構文解析規則に基づく手法では，与えられる構文解析規則が固定されているため，さまざまな言語現象に対処するのが困難である．そのため，上記の手法で対応できない場合には，構文解析の精度が下がり，構文解析システムの質が低下してしまう．

さまざまな言語現象に対処するために機械翻訳の分野では用例に基づく手法が試みられている[3][4][5]．同様に，構文解析の分野でも用例に基づく手法を用いた構文解析システムについて提案されている[6]．用例に基づく手法では，自動的に対象に適応するので，構文解析の精度が上昇し，構文解析システムの質を向上させることができる．しかしながら，このような手法に基づくシステムの性能を向上させるためには膨大な構文解析例を必要とする．

この問題を解決するために，比較的少量の構文解析例を用いて精度の高い構文解析を行える手法を提案する．構文解析例からより多くの構文解析規則を抽出することができれば，少量の構文解析例でも精度の高い構文解析が可能となる．そこで，本稿では，構文解析例（単語列と構文解析結果の組）からの帰納的学習を用いた構文解析手法を提案する．本手法では，構文解析規則は帰納的学習を用いて獲得され，さらに獲得された構文解析規則から再帰的に構文解析規則を獲得する．このように，自動的に構文解析規則を獲得することによりさまざまな抽象度の構文解析規則を獲得でき，それらを用いて最適な抽象度，つまりは具体的な規則から順に構文解析規則を適用することにより解析精度を向上させる．

本手法の利点としては，規則を人手により与えていないので，会話文などの非文法的な文や省略の多い文にも柔軟に対処できることが考えられる．また，一般的には構文解析を行うのに困難である対象に依存した文の解析も，システムが動的に適応することにより解決できることが多い．これは，このような限定された対象にのみ有効な構文解析規則を学習により自動的に生成することができるからである．さらには，構文解析例から学習できるので，適応能力が高く，さまざまな文章に対応できる．特に，本手法の学習アルゴリズムは言語に依存していないので，学習データさえあれば同一のアルゴリズム（帰納的学習）でどのような言語にも適用できる．すなわち，多言語対応構文解析手法としての能力を有するものと考えられる．

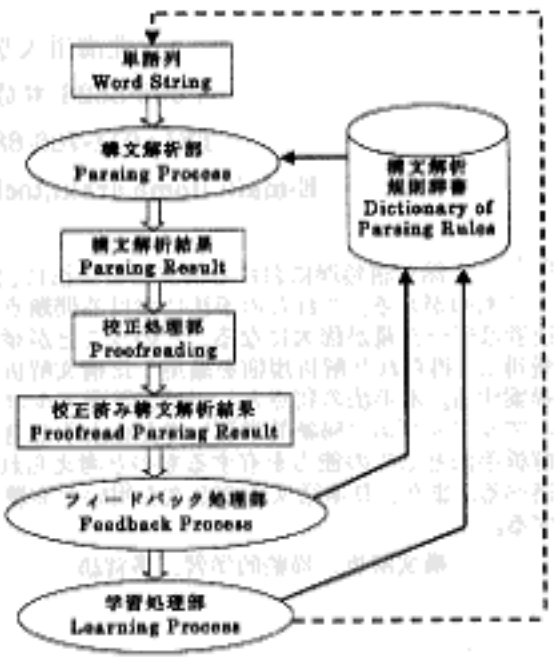

図 1 処理過程
Figure 1: Process

2 概要

図 1 に本手法の概要を示す．本手法に基づく実験システムは日本語文の構文解析を対象として作成されている．

最初に，日本語文の単語列を入力する．この単語列は帰納的学習による単語の認識[7]によって得られる．表 1 に入力する単語列の例を示す．

次に，構文解析部では獲得された構文解析規則辞書を用いて構文解析を行う．図 2 に構文解析結果の例を示す．また，表 2 に本手法における構文解析結果の内部表現の例を示す．表 2 の内部表現は図 2 の構文解析結果を表したものである．ここで，本手法で用いた内部表現の各記号の意味を説明する．“[ ]” で囲まれた範囲が文を表し，

表 1: 単語列の例
Table 1: Example of a Japanese word string

文	私は多くの研究を行った。						
単語列	私	は	多く	の	研究	を	行った。

表 2: 構文解析結果の内部表現例
Table 2: Example of representation of parsing result in this system

構文解析結果の内部表現	[(私 n は p)#3 (多く n の p)#1 $1(研究 n を p)#1 $1$3(行った v。)]

[ ]: 文, ( ): 文節, #, $: 係り受け関係, n: 名詞, p: 助詞, v: 動詞

図 2: 構文解析結果例

Figure 2: Example of parsing result

"( )" で囲まれた部分が文節を表す. 単語の後ろに付属しているアルファベットは, その単語の品詞を表している. さらに, "#" と "$" はそれぞれ係り元と係り先の文節に付属し, 付けられている番号はそれらの文節間の距離を表す. 表 2 において, 文節"(私 n は p)" は文節"(行った v.)" に係り, 文節 "(多く n の p)" は文節 "(研究 n を p)" に係り, そして, 文節 "(研究 n を p)" が文節 "(行った v.)" に係っていることを表している.

さらに, 校正処理部では人手により構文解析結果に誤りが含まれている場合に校正が施され, 校正済み構文解析結果が得られる.

フィードバック処理部では, 構文解析部より出力された構文解析結果に誤りが含まれている場合に, 構文解析結果と校正済み構文解析結果の差分より誤りの原因となった構文解析規則を推定することによりフィードバック処理を行う.

学習処理部では, 得られた校正済み構文解析結果と単語列の組を用いて帰納的学習を行い, 構文解析規則を獲得する. 表 3 に獲得される構文解析規則の例を示す. 表 3 における文についての構文解析規則で, ":" の左側にあるのが単語列で, 右側にあるのは構文解析結果の本手法における内部表現である. また, 係り受け関係および単語についての構文解析規則では, ":" の右側にある語についての品詞情報を付加したものが, 左側に表される. ここで, "@" は変数を表し, 直後に付属している数字はその規則中の変数の数を表している.

以上のような処理を繰り返し実行することにより, 構文解析規則辞書が充実し, この辞書を用いて行う構文解析の精度が上昇する.

以下では, 各処理の詳細について述べる.

2.1 構文解析部

構文解析部において, システムは構文解析規則を用いて入力された単語列の構文解析を行う. 入力に対して最適な構文解析規則を適用することにより, 構文解析結果が導き出される. 図 3 に構文解析部での処理例を示す. この図では, まず, (1)に示される単語列が入力され, 構文解析辞書中に(2)に示される(a), (b), (c)の構文解析規則がすでに獲得されていたと仮定する. ここで, 構文解析規則(a)中の "*" は係り受け関係の変数を表す記号である. このとき, (3)で示されるように構文解析規則の適用が行われる. 最初に, (a)の構文解析規則が適用され, 次に(a)の構文解析規

表 3: 構文解析規則の例

Table 3: Examples of parsing rules

文規則	<私 は 多く の 研究 を 行った 。: [(私 n は p)#3 (多く n の p)#1 $1(研究 n を p)#1 $1$3(行った v。)]> <@0 は 多く の @1 を @2 。: [(@0 は p)#3 (多く n の p)#1 $1(@1 を p)#1 $1$3(@2。)]>
係り受け規則	/(私 は) (行った 。): (私 n は p) (行った v。)/ /(多く の) (研究 を): (多く n の p) (研究 n を p)/ /(研究 を) (行った 。): (研究 n を p) (行った v。)/ /(@0 は) (行った 。): (@0 は p) (行った v。)/ /(多く の) (@0 を): (多く n の p) (@0 を p)/
単語規則	<私 : 私 n > <研究 : 研究 n > <行った : 行った v >

[ ]: 文, ( ): 文節, #,$: 係り受け関係, n: 名詞, p: 助詞, v: 動詞, @: 変数

(1) 入力単語列:
私 は 多く の 研究 を 行った 。
(I did a lot of researches.)
(2) 構文解析規則:
(a) <@0 は 多く の @1 を @2 。: [(@0 は p)*0 (多く n の p)#1
$1(@1 を p)#1 $1(@2。)]>
(b) /(私 は) (行った 。): (私 n は p) (行った v。)/
(c) <私 : 私 n > <研究 : 研究 n > <行った : 行った v >
(3) 構文解析規則の適用:
- (a) と (c)
<私 は 多く の 研究 を 行った 。 : [(私 n は p)*0 (多く n の p)#1
$1(研究 n を p)#1 $1(行った v。)]>
- (b)
<私 は 多く の 研究 を 行った 。 : [(私 n は p)#3 (多く n の p)#1
$1(研究 n を p)#1 $1$3(行った v。)]>
(4) 構文解析結果
[(私 n は p)#3 (多く n の p)#1 $1(研究 n を p)#1 $1$3(行った v。)]

図 3: 構文解析処理の例

Figure 3: Example of parsing process

則中の変数に(c)の構文解析規則が適用される．最後に，係り受け関係の変数があるので，これに対して(b)の構文解析規則を適用することにより係り受け関係が明確になる．このようにして，(4)に示される構文解析結果が得られる．

複数の構文解析規則が適用可能であるときは，次のような順序で構文解析規則を適用する．

① 最長一致する構文解析規則で字面が多いもの．

② 最も少ない構文解析規則で構成されているもの．

③ 構成している構文解析規則の正解析率の平均が最大のもの．

④ 構成している構文解析規則の正解析度数の和が最大のもの．

⑤ 構成している構文解析規則の誤解析度数の和が最小のもの．

⑥ 構文解析規則辞書の登録順が早いもの.

2.2 フィードバック処理部

フィードバック処理部では,構文解析部より出力された構文解析結果に誤りが含まれている場合に,構文解析結果と校正済み構文解析結果の差分より誤りの原因となった構文解析規則を推定することによりフィードバック処理を行う.

以下の手順に従って,フィードバック処理が施される.

① 単語列および校正済み構文解析結果の共通部分と差異部分を決定する.

② 共通部分を正解析,差異部分を誤解析とする.

③ 正しく解析した構文解析規則の正変換度数を1増加させる.

④ 誤って解析した構文解析規則の誤変換度数を1増加させる.

2.3 学習処理部

学習処理部では,単語列と校正済み構文解析結果の組を用いて帰納的学習を行い,構文解析規則を獲得する.構文解析規則は単語列と校正済み構文解析結果の共通部分と差異部分から獲得される.この操作を獲得された構文解析規則について,新たな構文解析規則が抽出されなくなるまで再帰的に繰り返す[8].係り受け関係については,単語間の複数の関係がすべて同一の場合のみ共通部分であるとし,それ以外は差異部分とする.係受け関係の差異部分から獲得される構文解析規則は係受けの語の組にして記憶する.構文解析規則は,係受け関係の共通部分が 60%以上で,かつ字面上の一致が 50%以上の場合に獲得される.

学習処理部での帰納的学習を用いた構文解析規則の獲得例を図4に示す.ここでは,(1)に示す(a)～(d)の構文解析規則がすでに獲得されていると仮定する.ここで,(a)と(b)の共通部分と差異部分(下線部)から(e)のような差異部分が変数化された構文解析規則が抽出される.また,(f)と(g)のような変数化された単語の構文解析規則が抽出される.同じように,(c)と(d)の係り受け関係の構文解析規則からも,共通部分と差異部分(下線部)の関係から,(h)のような差異部分が変数化された構文解析規則として抽出される.また,(i)と(j)のような変数化された単語の構文解析規則も抽出される.

3 実験

本手法の性能を評価する実験を行った.実験で

(1) 獲得されていた構文解析規則:
(a) <@0 は 研究 を 行った 。: [(@0 は p)#2 (研究 n を p)#1 $1$2(行った v。)]>
(b) <@0 は 活動 を 行った 。: [(@0 は p)#2 (活動 n を p)#1 $1$2(行った v。)]>
(c) /(私 は) (行った 。): (私 n は p) (行った v。)/
(d) /(彼 は) (行った 。): (彼 n は p) (行った v。)/
(2) 抽出された構文解析規則
(a) と (b) より
(e) <@0 は #1 を 行った 。: [(@0 は p)#2 (#1 を p)#1 $1$2(行った v。)]>
(f) <研究 : 研究 n >
(g) <活動 : 活動 n >
(c) と (d) より
(h) /(@0 は) (行った 。): (@0 は p) (行った v。)/
(i) <私 : 私 n >
(j) <彼 : 彼 n >

図4: 構文解析規則の再帰的抽出例

Figure 4: Example of the extracted parsing rules from the acquired parsing rules

はEDR日本語コーパス[9]中の5,000文を用いた．構文解析の結果については，係り受け関係にのみ着目した．係り受け関係の正解については，EDR日本語コーパスの係り受け関係と一致するものとした．

3.1 実験手順

初期状態では構文解析規則辞書は空である．以下の手順により実験を行った．

① 日本語文を単語列に変換する．

② 単語列を入力する．

③ 入力された単語列について構文解析規則を用いた構文解析を行い，構文解析結果を得る．

④ 構文解析結果とEDR日本語コーパスから得られる校正済み構文解析結果からフィードバック処理を行う．

⑤ 単語列と校正済み構文解析結果の組から帰納的学習を用いて構文解析規則を獲得する．

⑥ ⑤で獲得された構文解析規則とすでに獲得された構文解析規則から構文解析規則を抽出する．

⑦ ⑥を新しい規則が抽出されなくなるまで再帰的に繰り返す．

⑧ 以上の操作を日本語文 5,000 文について繰り返し行う．

ここで，手順⑤で単語列と校正済みの構文解析結果の組について帰納的学習を行うのは，処理時間の問題から，入力した文の前 100 文について行った．

3.2 実験結果

表5に実験結果を示す．日本語文5,000文の係り受け関係についての正解率は22.7%であった．表5の結果を図5のようにグラフで表す．この図は 500 文毎の係り受け関係の正解率の変化を表したものである．また，図6に獲得された構文解析規則数を示す．

表 4: 入力文数中の係り受け数と正解析数
Table 4: Numbers of modifications and correct parsing rate in input sentences

文数	係り受け数	正解析数
1～500	5638	335
501～1000	6317	819
1001～1500	6514	1289
1501～2000	7711	1768
2001～2500	7711	2230
2501～3000	5683	1592
3001～3500	6312	1940
3501～4000	5571	1499
4001～4500	5971	2058
4501～5000	6065	2058
1～5000	63493	14439

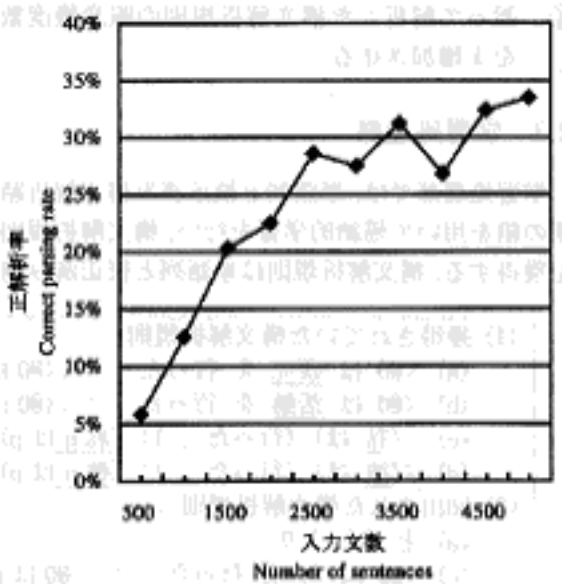

図 5: 入力文数による正解析率の変化
Figure 5: Change of correct parsing rate

4 考察

図 5 の実験結果のグラフから本手法に基づく構文解析結果の精度が徐々に向上しているのが認められる．初期状態では構文解析規則辞書が空の状態であるので，入力文数が少ない段階では入

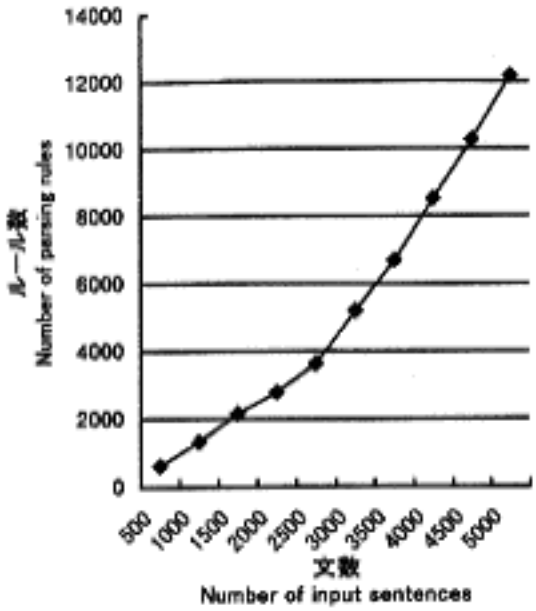

図 6: 入力文数による構文解析規則数
Figure 6: Number of parsing rules

力単語列に対して適用できる構文解析規則が構文解析規則辞書中に無いことが多い．そのため，構文解析部で係り受け関係を決定することができない解析不能となり，入力文数の少ない状態では正解析率は低くなる．入力文数が増加すると，図 6 で示されるように構文解析規則数が増加し，それに伴い入力単語列に対して適用できる構文解析規則も増加する．これにより，構文解析部で係り受け関係を決定することができ，正解析率も上昇する．したがって，本手法には係り受け関係について徐々に精度を向上させる能力があり，その有効性を見出すことができる．

次に，実験結果での誤りについての例を挙げる．入力単語列に獲得された係り受け関係についての構文解析規則を適用する場合に，係り受け関係がないものについても構文解析規則を適用してしまい誤りが起こった．特に変数が含まれる場合に誤りが起こりやすい．これは，できる限り構文解析規則を適用したために起こったものと考えられる．しかしながら，この誤りについては構文解析規則辞書が充実することによって，次第に減少していくと考えられる．

構文解析規則数が増加すると，新たな問題が起こる．それは，処理時間の増大である．これは，構文解析処理および構文解析規則抽出処理の時間が増大するということである．この問題を解決するための方法を考える．構文解析処理については，辞書の階層化などを行うことにより，適用する構文解析規則辞書を見つける時間を減少させることが考えられる．構文解析規則抽出処理については，ヒューリスティクスを追加することにより，より効率的に構文解析規則を抽出することが考えられる．

5 おわりに

与えられた構文解析規則を用いた構文解析手法と用例に基づく構文解析手法の問題点を解消するために，本稿では，実例からの帰納的学習を用いた構文解析手法を提案した．さらに，提案した構文解析手法に基づくシステムを作成した．作成したシステムでは，構文解析規則は帰納的学習を用いて獲得し，さらに獲得された構文解析規則から再帰的に構文解析規則を獲得した．これにより，自動的に構文解析規則を獲得し，さまざまな抽象度の構文解析規則を獲得できた．獲得された構文解析を用いて最適な抽象度，つまりは具体的な規則から順に構文解析規則を適用することにより係り受け解析実験を行った．実験で用いた日本語文は EDR コーパス中の 5,000 文である．実験結果より，係り受け関係の正解析率が徐々に上昇することを確認した．したがって，本手法には，本手法に基づくシステムの精度を徐々に向上させる能力があり，その有効性を見出した．

今後の課題としては，さらに精度の向上を図るために，ヒューリスティクスの追加や辞書の階層化などが挙げられる．また，より多くのデータを用いた実験を予定している．

謝辞

本研究の一部は文部省科学研究費補助金(課題番号 10680367)により行われた．

参考文献

[1] E. Brill and P. Resnik: "A rule-based approach to prepositional phrase attachment disambiguation", Proceedings of the 15th COLING, pp. 1198-1204, 1994.

[2] U. Germann: "A deterministic dependency parser for japanese", Asia-Pacific Association for Machine Translation, pp. 547-555, 1999.

[3] V. Sadler and Vendelmans: "Pilot implementation of a bilingual knowledge bank, Proceedings of the 13th COLING, pp. 449-451, 1990.

[4] C. Stanfill and D. Waltz: "Toward memory-based reasoning", Communications of the ACM, Vol. 29, No. 12, pp. 1213-1228, 1986.

[5] S. Sato and M. Nagao: "Toward memory-based translation", Proceedings of the 13th COLING, pp. 247-252, 1990.

[6] M. H. Al-Adhaileh and T. E. Kong: "A flexible example-based parser based on the SSTC", Proceeding of the ACL, pp. 687-693, 1998.

[7] 荒木健治, 栃内香次: "帰納的学習による語の獲得および確実性を用いた語の認識", 電子通信学会論文誌 D-II, Vol. J75, No. 7, pp. 1213-1221, 1992.

[8] 荒木健治, 高橋祐治, 桃内佳雄, 栃内香次: "帰納的学習を用いたべた書き文のかな漢字変換", 電子通信学会論文誌 D-II, Vol. J79, No. 3, pp. 391-402, 1996.

[9] Japan Electronic Dictionary Research Institute Ltd.: "The EDR Electronic Dictionary Technical Guide", 1995.