修 士 論 文

コーパスにおける文脈情報を利用した文法開発支援

指導教官　奥村学 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

川口恭伸

1997 年 2 月 14 日

要旨

本論文では、既存の文法に対し、特定のコーパスに対するカバレッジをあげるために追加すべき規則の仮説を生成し、既存の文法で解析可能な部分における局所文脈情報を用いて、仮説選択のためのスコアリングを行なう方法を提案する。

実用の自然言語処理システムに組み込んで, 実用に耐え得る文法を構築することは困難かつ膨大なマンパワーが必要な作業である. また, あらゆる分野に適用可能で, 過生成でないような文法を作ることは事実上不可能である. そこで, 既存の文法を特定のコーパスに対して修正する作業が必要である.

この作業において人間の作業負荷を軽減するために, 非文解析における誤り訂正のための仮説生成・仮説選択のアプローチと同じように, 解析不能な文を解析可能にするための尤もらしい文法仮説を提示する方法が有効であると考えられる. 本研究で採用した仮説提示の手法は、次のような手順による。(1) 解析できない文に対して, 部分解析結果を手がかりにして現在の文法の欠落を仮定し, それを補う文法仮説を生成する. (2) コーパスベースの統計的評価値を用いて, 仮説を絞り込む. (3) 人間が言語直観に基づいて, 規則として導入する仮説を決定する.

関連する研究として, 清野らによる仮説選択の研究と大谷らによる文法の半自動修正の研究がある. 前者はより小さく, 他の有力な仮説と競合しないものを優先する評価値によって仮説選択を行ない, 後者は規則の型に関するヒューリスティックスを用いて仮説生成の時点で仮説数を絞り込んだ. これに対し、我々は局所文脈情報を用いた。

局所文脈情報とは、あるカテゴリの直前直後にくるカテゴリのことである。我々の評価尺度は仮説の局所文脈情報と、同じラベルをもつカテゴリの正解構文木集合のなかでの局所文脈情報から、その仮説の尤もらしさを決定する。

実験において、上位 10 仮説に絞った場合には初期文法で解析できなかった文のうち、81.4% に対して正解を含む仮説を提示できることが確認された。上位 1 仮説に絞った場合には、解析できない文のうち 42.3% に対して正解仮説を提示することができた。

目 次

第 1 章

序論

実用の自然言語処理システムに組み込んで、実用に耐え得る文法を構築することは困難かつ膨大なマンパワーが必要な作業である。これまで言語処理システムに実装された文法は、人間の手による膨大なトライ&エラーの繰り返しによって作成されてきた。

近年、EDR[9] の辞書やコーパスをはじめとして、大規模な機械可読な言語データの整備が進み、大規模なデータを扱うための計算機能力が向上したことから、大規模で多様な言語使用を含むコーパスを使用した様々な言語処理の研究が盛んになった。性能のよい文法とは、対象領域の文を正しく解析でき、解析結果に曖昧性が少なく、必要な規則数も必要最小限に抑えられているような文法のことを指す。大規模コーパスの出現と、様々な言語資源の整備にしたがって、人間の手による膨大なトライ&エラーの繰り返しによって作成されてきたこれまでの文法開発手法は困難になりつつある。

自然言語処理に使用するための様々な言語知識を大規模コーパスから自動的に獲得する研究が盛んに行なわれている。自動獲得された文法は、人間の手による文法に比べ、人間の主観に影響されにくい、文法作成のためのコストが少なくてすむ、統計情報を付加することが容易である、というような利点がある [10] が、これまでに構築された様々な言語知識や人間の主観的な判断から独立して獲得される文法は、人間の言語直観との不一致や既存の言語知識との互換性の面で問題がある。

そこで、自動獲得した文法ないし人間の手によって作成された文法を、対象領域のコーパスにあわせて適宜修正を加えるという作業が不可欠なのであるが、文法のある部分を、他の部分との整合性をとりつつ修正するという作業は、文法の規模が拡大するにしたがって困難を極めることになる。そこで、計算機を用いてこの作業を支援することは有用だと

いえる。

文法修正の作業は、3 つに分類することができる。1 つは、欠落した文法規則を推定して追加する**規則追加**。これは文法の対象領域に対するカバレッジ (適用範囲) を向上させるが、逆に文法による解析結果の曖昧性が増加する可能性がある。2 つめは、冗長な文法規則の削除を行なう**規則削除**。これは解析結果の曖昧性を抑えることができるが、削除する規則によってはカバレッジを下げる可能性がある。そして 3 つめは粒度の粗いカテゴリをより精密に言語使用と対応するいくつかのカテゴリに分割する**規則分割**である。これは、曖昧性を下げることができるが、カテゴリが細分化され、文法規則数が増加してしまう可能性がある。

このなかで、もっとも容易であるのは**規則追加**の支援である。これは、規則追加によっては既に解析できている文が解析できなくなるといった致命的な悪影響が発生しないことと、文法の自動獲得で用いられている評価基準を参考にすることができるからである。**規則削除**は削除する規則によって解析不能になる文がありうるが、文法規則とそれを必要とする構文木の対応関係を把握しておけば、これを未然に防ぐことができるし、自動的に冗長な規則を発見して削除することは可能である [10]。**規則分割**は元となる規則の集合が、分割するとどちらかに継承されるのか、それとも両方に継承されるのかということを決定しなくてはならないが、これは難しい作業であると思われる。また既に解析できている文が解析できなくなるという悪影響も起こり得るので、複雑な規則間の関係を考慮する必要がある。しかし、文法による解析結果の曖昧性を下げ、文法をより精密なものにするためには不可欠な作業である。

我々は、これら 3 つの分類のうち、**規則追加**に関する支援について研究を行なう。

本研究の目標は、追加すべき規則の尤もらしい仮説を開発作業者に提示することである。それにはまず、解析不可能な文における部分解析結果と現在の文法をもとに、1 つの規則追加で解析が可能になるような規則の仮説を集め、その後で、尤もらしさを量る統計的評価値を用いて仮説の順位付けを行ない、上位いくつかの仮説を、開発作業者に提示する候補とする。

仮説生成の方法については、既に提案されている規則ベースの方法 [3] を用いる。この方法は数多くの無駄な仮説も生成してしまう。

生成された仮説の尤もらしさを量る統計的評価値としては、コーパスからの自動文法獲得の研究 [1][7][8] において品詞列からカテゴリを推定するための統計データとして用いら

れている局所文脈情報を用いた。仮説のカテゴリの局所文脈情報と、現在の文法で正しく解析できる正解構文木の集合におけるそのカテゴリの局所文脈情報を比較し、正解構文木集合の中で出現する可能性が高いほど, その仮説は尤もらしいと判断されるべきだというヒューリスティックを用いた。

本論文の構成は以下の通りである。まず第 2 章で、仮説生成の方法と、仮説選択の必要性と他の研究における仮説選択について述べる。そして第 3 章では、本研究で提案する環境の分布を用いた仮説選択の方法について述べ、第 4 章でこの仮説選択法の有効性を調査するために行なった実験とその結果について述べる。最後に、第 5 章で本研究の結論と今後の課題を述べる。

第 2 章

解析失敗からの仮説生成および仮説選択

この章では、まず、解析失敗からの仮説生成の方法について述べ、それから仮説選択の必要性と他の研究における仮説選択の方法について述べる。

2.1 文法開発

2.1.1 文法開発と文法開発支援環境

本研究で想定する**文法開発**とは、既存の文法に対し、特定のコーパスに対応するよう修正を加え、その性能をあげる作業のことである。

また、この作業を支援するための計算機環境を、**文法開発支援環境**と呼ぶ。

文法開発支援環境を通して文法開発の作業をする人間のことを**開発作業者**、もしくは単に**ユーザー**と呼ぶ。以後、本論文で単にユーザーという場合は、開発作業者のことを指す。

2.1.2 文法形式

本研究では、扱う文法のクラスを文脈自由句構造文法 (Phrase Structure Context Free Grammar;**PSCFG**) とした。

PS-CFG における文法規則は、一般的に以下のような形式で記述される。

$$g : A \rightarrow \alpha_1, ..., \alpha_n$$
$$(A \in \{V\}, \alpha_k \in \{V \cup T\})$$

Vは非終端記号の集合、Tは語彙カテゴリ (終端記号) の集合である。そこで、**規則の欠落**を補うための文法仮説 h を以下のように定義する。

$$h : A \rightarrow \alpha_1, ..., \alpha_n$$
$$(h \notin G, A \in \{V\}, \alpha_k \in \{V \cup T\})$$

大文字の G は小文字の gの集合を表している。

G に現れない A とα_kの関係全てが仮説になりうるため、もし、ある文法の規則の欠落を見つけ出そうとしても、その探索空間は非常に広い。そこで、本研究では、現在ある文法に対して規則を一つ追加するだけで解析が成功する仮説のみを扱う。

2.1.3 一般仮説とインスタンス仮説

仮説は、その適用範囲から、2 つのレイヤーに分けられる。

- 一般仮説 (文法仮説)
  文法の欠落を補う CFG 規則の仮説
- インスタンス仮説 (不活性弧仮説)
  各文におけるカテゴリの欠落を補う仮説

一般仮説は、現在の文法で欠落していると推測される文法規則の仮説であり、解析失敗した各文において生成されるインスタンス仮説を一般化したものである。インスタンス仮説は文の情報、文の内部での位置情報など、属する文の情報を持っているが、一般仮説は個別の文情報を持たない CFG 形式の規則仮説である。

本研究ではこのうち不活性弧仮説のみを扱った。コーパス全体における一般仮説の仮説選択については扱っていない。

2.2 文法開発の手順

文法開発は、まず例文を現在の文法で構文解析し、解析が失敗した場合にはシステムが解析成功に導くために仮説を生成し、評価尺度を用いて仮説を絞り込み、最後に人間が導入すべき仮説を決定する, という手順 (図 2.1) で行なわれることを想定している。

文法開発支援システムは以下の 2 つのモジュールから構成される (図 2.2)。

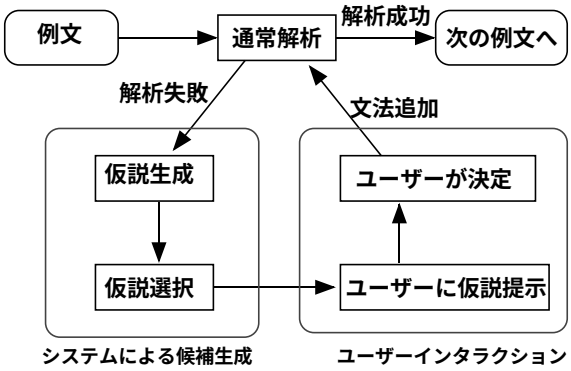

図 2.1: 文法開発の手順

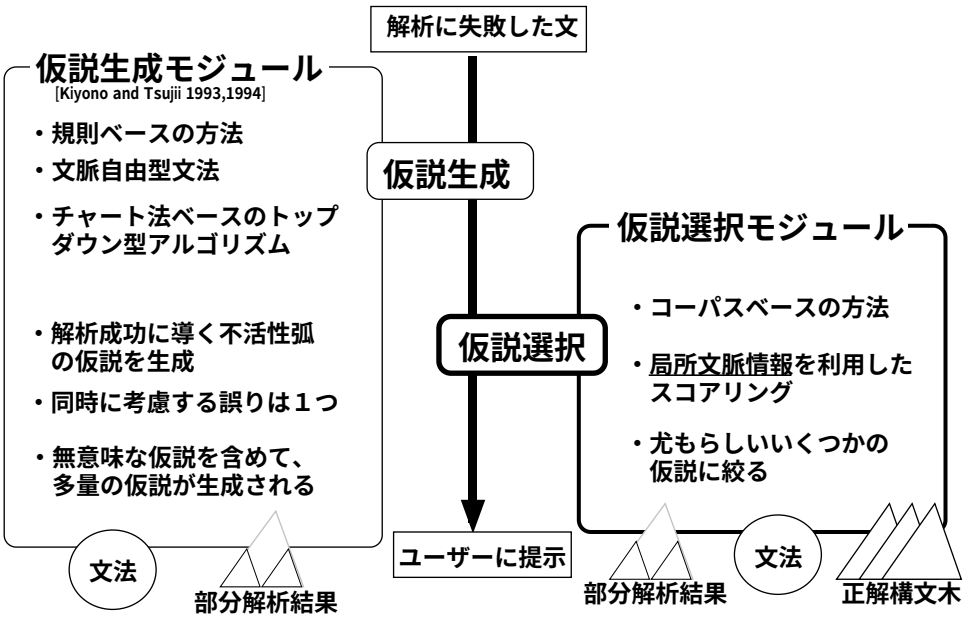

図 2.2: システムの処理

仮説生成モジュール

清野ら [3] に基づくチャート法ベースの仮説生成アルゴリズムを用いた。まず、現在の文法からボトムアップに生成できる不活性弧はすべて生成しておき、それから、生成された不活性弧をもとに, 以下のアルゴリズムをトップダウンに適用して仮説を生成する。

仮説生成アルゴリズム [3]: 単語位置 x_0から x_1における、ラベル A の不活性弧の仮説 [hypo(A):x_0,x_n] は、以下のようなステップによる仮説生成によって導くことができる。

1. 単語位置 x_0 から x_n までを覆う $[ie(B_1) : x_0, x_1], ..., [ie(B_n) : x_{n-1}, x_n]$ なる不活性弧列のそれぞれにおいて、$A \rightarrow B_1, ..., B_n$ という新たな規則を仮定し、不活性弧仮説 $[hypo(A) : x_0, x_n]$ を生成する。

2. 既に存在する $A \rightarrow A_1, ..., A_n$ という形の規則それぞれに対し、$[ie(A_1) : x_0, x_1], ..., [ie(A_{i-1}) : x_{i-2}, x_{i-1}], [ie(A_{i+1}) : x_i, x_{i+1}], ..., [ie(A_n) : x_{n-1}, x_n]$ なる不完全な不活性弧列[1] があるならば、$[ie(A_i) : x_{i-1}, x_i]$ に関してこのアルゴリズムを呼び出す。

このアルゴリズムは、現在存在しない文法規則を一つだけ仮定することで解析が成功するような不活性弧の仮説を出力する。

トップノードの予測から開始するので、初期値として、文の開始位置、終了位置、トップノードカテゴリを与える必要がある。

仮説選択モジュール

仮説生成モジュールでは大量の仮説が生成されるが, その殆んどは正解になりえないものである。仮説選択モジュールは, 仮説の前後の部分解析結果と、正解構文木集合における同カテゴリの局所文脈情報を手がかりに仮説の尤もらしさをスコアリングし、仮説の絞り込みを行ない、ユーザーに提示すべき仮説を選択する。正解構文木集合とは、現在の文法で正しい解析が行なわれた文の構文木の集合である。

2.3 仮説選択

前節の方法で生成された仮説は、正しく文法の欠落を回復する仮説の他に、数多くの不必要な仮説を含んでおり、すべての仮説を文法開発者に提示するのは、多くの無駄な作業

[1] $[ie(A_i) : x_{i-1}, x_i]$ に相当する不活性弧が不足していると考える.

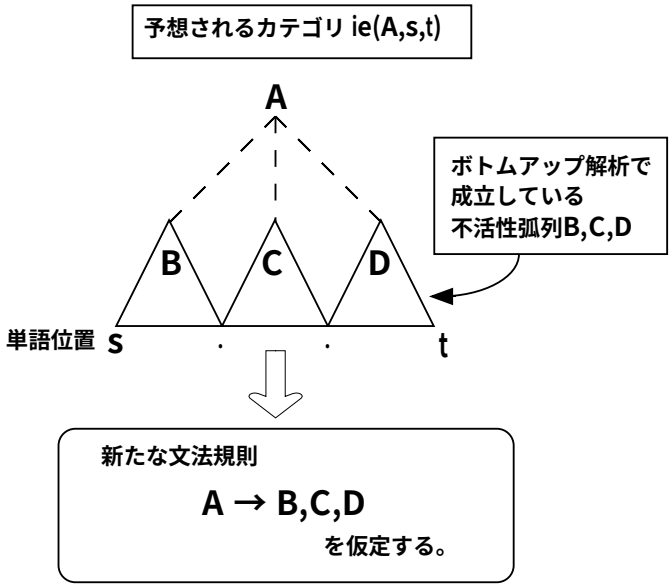

図 2.3: 仮説生成アルゴリズム-ステップ 1

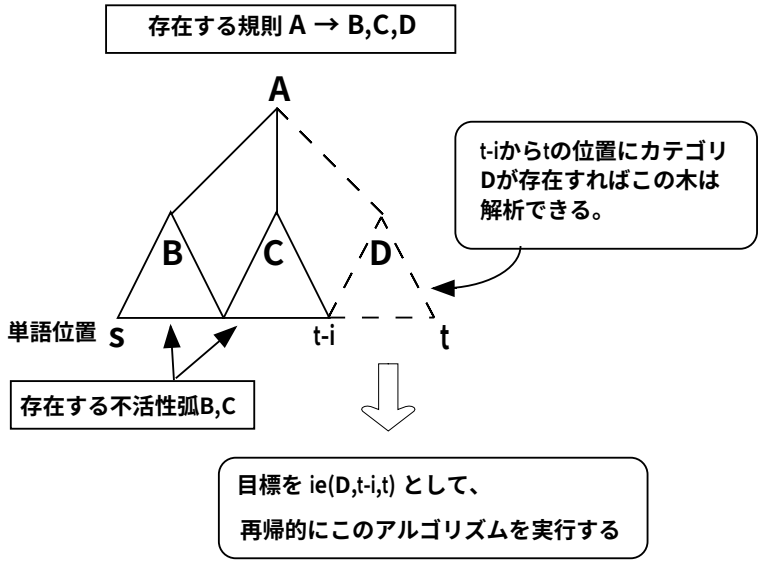

図 2.4: 仮説生成アルゴリズム-ステップ 2

を開発者に強要することになる。そこで、なんらかの評価尺度を用いて仮説を選択することが必要である。仮説選択の研究には、大谷ら [5][6] による研究と、Kiyono ら [3][4] による研究がある。

2.3.1 Kiyono らの方法

Kiyono らの研究 [3][4] では、仮説集合の持つ情報のみを用いて仮説の尤もらしさを決定する。これは、以下の２つの仮定

- 基とする文法は、一般的な言語現象ならば解析できる、十分な適用範囲を持つ。
- 対象とするコーパスは一般的な言語からは外れた部分言語であり、あまり多くの例文を持たない小量のコーパスである。しかし、そこでの言語使用は比較的限定されている。

に基づいて、文法開発の目標を語彙知識などのローカルな言語知識の獲得としているためである。

具体的には、以下のようなステップで仮説選択が行なわれる。

1. 各文において、仮説間の関係として、同時に成立しうる補完関係と、ある子カテゴリに対して競合を起こすために同時に成立し得ない競合関係のどちらかに分け、その関係を表す AND–OR グラフを作る。
2. グラフの各項の初期値を、各項の文法仮説が持つ部分木をもとに計算する。それぞれの部分木がより小さい方が初期値が高くなる。
3. 競合関係の仮説どうしは値を弱めあい、補完関係は高めあうようにして再計算を行ない、閾値より値の小さくなったものは削除する。仮説が削除される頻度が少なくなるまで再計算を行ない、残った仮説を候補とする。

仮説の AND–OR グラフ表現

Kiyono らの仮説選択の基本概念のひとつは、

> ある仮説は、他の競合仮説のなかに尤もらしいものがない場合にのみ、尤もらしい仮説であると判断されるべきである

というものである。この概念に基づく仮説選択を実現するために、仮説間に競合関係および補完関係を定義する。

同じ文から生成される仮説間には、競合関係もしくは補完関係のどちらかの関係が成立している。

競合関係 同じ不活性弧を修復させるために生成される仮説間の関係であり、この中の 1 つだけが正しい文の構造と一致している。

補完関係 同じ文の中で、2 つの異なる部分から生成される仮説は補完関係にあるという。

さらに、Kiyono らの仮説選択法では仮説間の関係を AND-OR グラフで保持する。AND ノードおよび OR ノードはそれぞれ、補完関係と競合関係を表現する。

インスタンス仮説の評価値 LP

Kiyono らの仮説選択の基本概念のもうひとつは、

> 現在の文法が十分に広範囲 (comprehensive) であると仮定すると、一般的な書き換え規則よりも語彙や連語の知識の方がより尤もらしくなるべきである。

というものである。このような仮説選択を実現するために、各インスタンス仮説の尤もらしさを表す LP 値 (Local Plausibility) は以下のように、構成する部分木の大きさに反比例するように設定される (図 2.5)。

$$LP(Hypo_i) = 1 - \frac{W(Hypo_i) \times H(Hypo_i)}{W(S) \times H(S)}$$

一般仮説の評価値 GP

インスタンス仮説の LP 値をすべて求めたら、つぎは一般仮説の GP 値 (Global Plausibility) を定義する。一般化仮説の尤もらしさは、インスタンス仮説のなかのどれかひとつが正解であることの尤もらしさとして定義されている。

$$GP(HP) = 1 - \prod_{i=1}^{n}(1 - LP(HP_i))$$

インスタンス仮説の数が増えると、GP(HP) 値はより 1 に近づく。

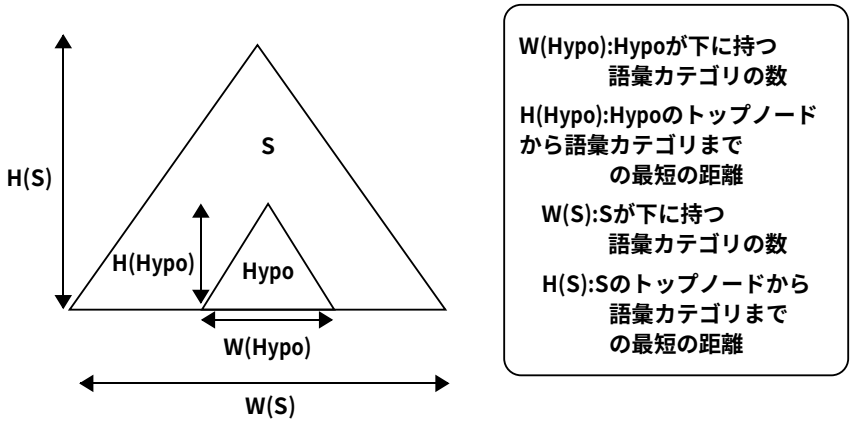

図 2.5: LP 値の初期値

AND–OR グラフを用いた LP と GP の反復的計算

正解のインスタンス仮説が仮説グラフ中に存在するという仮定に基づいて, 各文の仮説 AND–OR グラフにおいて LP 値の再計算が行なわれる。この仮定から、まずグラフのトップノードには 1 が割り当てられる。そして、各ノードの GP 値に比例して、親ノードの LP 値が子ノードに分配される。

しかし、前項の GP 値の定義では仮説 AND–OR グラフの中間ノードに関する GP 値は定義されていないので、葉ノードの GP 値からボトムアップに中間ノードの GP 値を定義する。

OR ノードの GP 値 少なくともひとつの子ノードが正解である尤もらしさ

$$GP(OR) = 1 - \prod_{i=1}^{m}(1 - GP(Node_i))$$

AND ノードの GP 値 子ノードのすべてが正解であるもっともらしさ

$$GP(AND) = 1 - \prod_{i=1}^{m} GP(Node_i)$$

各ノードの GP 値が計算できたら、トップノードに LP 値 1 を与え、子ノードへの LP 値の分配を行なう。

OR ノードの LP 値 子ノードは相互に排他的な仮説であるので、OR ノードの LP 値は子ノードの GP 値に比例して分配される。

$$LP(Node_i) = \frac{GP(Node_i)}{\prod_{j=1}^{m}(GP(Node_j))} \times LP(OR)$$

AND ノードの LP 値 子ノードに同じ値が分配される

$$LP(Node_i) = LP(AND)$$

結果的に各インスタンス仮説に新たな LP 値があたえられる。新たな LP 値は、各文において AND–OR グラフで表現された仮説間の関係と、コーパス全体における一般仮説の尤もらしさを反映した値になる。

各インスタンス仮説に新たな LP 値が与えられたら、再び GP 値を計算し、反復的にこれまでの計算を繰り返す。

また、ボトムアップに GP 値を計算する過程で、同じ OR ノードを親にもつノードの中で最大の GP 値を持つノードに比べて十分に小さな GP 値を持つノードは仮説 AND–OR グラフから削除する。

サイクル毎に削除されるインスタンス仮説の数をカウントし、一つのインスタンス仮説も削除されないサイクルがいくつか続いた場合に、計算プロセスが収束したとみなして停止する。このとき高い GP 値を持つ仮説は追加すべき新たな文法の最終候補として提示する。

2.3.2 大谷らの方法

大谷らの方法は、仮説生成のプロセスがそのまま仮説選択になっている。すなわち生成能力の異なる 3 つの仮説生成ステップを特定の順で適用し、あるステップで仮説が生成されたら残りのステップは実行しない、という方法で生成される仮説の数を制限している。

- まず、ボトムアップ解析を行ない、生成された部分解析木の**数が最小になる組合せ**を求める。それから、以下に示す方法を順に適用し、解析可能な規則を生成した時点で終了する (図 2.6)。

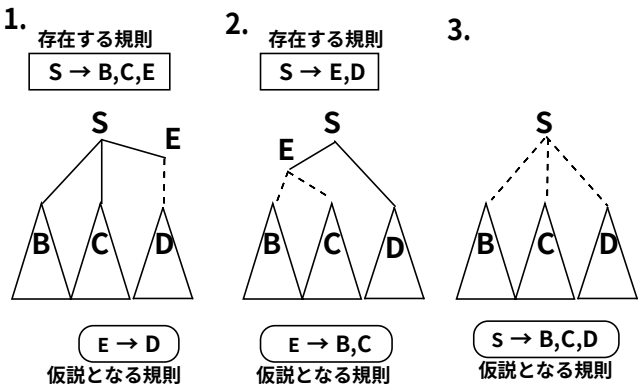

図 2.6: 大谷らの仮説生成法

1. ある部分解析木のカテゴリラベルを適当に変えて、入力文が受理できるならば、その変換規則 (Unary Rule) を生成する。
2. ある 2 つの部分解析木を繋げて適当な新しいカテゴリを生成することで、入力文が受理できるならば、その接合の規則 (Binary Rule) を生成する。
3. 部分解析木の最小の組合せを直接開始記号に繋げる。

1,2 は前章のステップ 2 に、3 はステップ 1 に似ているが、再帰せず、文全体に対して一度だけ適用する点が異なる。再帰しないため、仮説の生成能力は限定されているが、大谷らは音声対話システムに用いる文法を対象にしているため、長く複雑な構造を持つ文が出現する可能性が低いという対象領域の特性から、このような仮説生成が有効になっているものと考えられる。

第 3 章

文脈情報を用いた仮説選択

この章では、仮説選択モジュールにおいて、仮説を含む文の部分解析結果と正解構文木集合における局所文脈情報を用いて仮説をスコアリングする方法について述べる。

3.1 他の手法の問題点

前章で紹介したように、Kiyono らはより小さく, 他の有力な仮説と競合しない仮説を優先する評価値によって仮説選択を行ない, 大谷らは規則の型に関するヒューリスティックスを用いて仮説生成の時点で仮説数を絞り込んだ。しかし、これらは仮説の大きさや他の仮説との競合関係など、仮説の間接的な “良さ” を評価しているにすぎない。

これに対して、現在の文法で正しく解析できた解析結果における統計情報を利用すれば、不活性弧仮説のカテゴリが前後の文脈に対してどれくらい存在しやすいかを調べることができる。そこで、本研究ではこの情報を利用して仮説のスコアリングを行なった。

3.2 局所文脈情報を用いた仮説スコアリング

3.2.1 局所文脈情報

局所文脈情報 (local contextual information) とは、あるカテゴリの左右に出現するカテゴリのことである。同じラベルを持つカテゴリは局所文脈情報が類似しているという仮定から、タナラックらのコーパスからの自動文法獲得の研究 [1] において品詞列からカテゴ

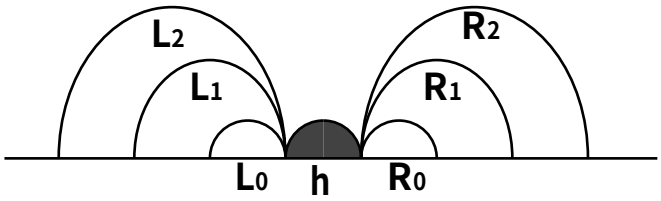

図 3.1: 左右連接カテゴリ

リを推定するための統計データとして用いられている。また、局所文脈情報という用語は用いていないが、Brill ら [7]、森ら [8] の自動文法獲得の研究においても、同様の情報が用いられている。

これらの研究では局所文脈情報として語彙カテゴリ (品詞) を用いているが、我々の仮説選択のタスクにおいては、仮説の左右には部分解析結果である不活性弧が存在する (図 3.1) ので、この情報も利用することができる。

もし、仮説の局所文脈情報として語彙カテゴリのみを考慮する場合、一つの仮説に対してラベルと局所文脈情報のセットは一つしか得られない。例えば、図 3.1の例では、得られる局所文脈情報は (L_0, h, R_0) のみである。得られた唯一の局所文脈情報が正解構文木集合内に現れなければ、出現頻度は 0 になってしまうので、データスパースネスの問題が出やすいと考えられる。

そこで語彙カテゴリだけでなく、左右に連接する非終端記号 (部分解析結果の不活性弧) も、局所文脈情報として利用するならば、より多くの局所文脈情報を得ることができ、この問題が回避できると考えられる。この場合、図 3.1の例で得られる局所文脈情報は (L_0, h, R_0)、(L_0, h, R_1)、(L_0, h, R_2)、(L_1, h, R_0)、(L_1, h, R_1)、(L_1, h, R_2)、(L_2, h, R_0)、(L_2, h, R_1)、(L_2, h, R_2) である。

3.2.2 局所文脈情報を用いた仮説の評価値

我々が提案する評価値は、部分解析木における仮説と文脈とのセットが正解構文木集合の中で出現する可能性が高いほど、その仮説は尤もらしいと判断されるべきだというヒューリスティックに基づいている。

まず、局所文脈情報として語彙カテゴリのみを考慮する場合の評価値 $Score_{lc}$[1]を定義する。次に、局所文脈情報として語彙カテゴリと非終端記号を共に考慮する場合の評価値 $Score_{lc+nt}$[2]を定義する。

評価値 $Score_{lc}$: 語彙カテゴリのみを考慮する場合

ラベル Cat_h を持つ 不活性弧仮説 h の評価値は、左右に接する語彙カテゴリを $L_0(h), R_0(h)$ とすると、
正解構文木集合における、$L_0(h), Cat_h, R_0(h)$ の出現頻度 $N(L_0(h), Cat_h, R_0(h))$ 、 文脈 $L_0(h), R_0(h)$ の出現頻度 $N(L_0(h), R_0(h))$ に関して以下のように定義する。

$$Score_{lc}(h) = \frac{N(L_0(h), Cat_h, R_0(h))}{N(L_0(h), R_0(h))}$$

評価値 $Score_{lc+nt}$: 語彙カテゴリと非終端記号を考慮する場合

不活性弧仮説 h の左右に接する語彙カテゴリおよび非終端記号 (図 3.1) をそれぞれ $L_0(h), ..., L_m(h)$、 $R_0(h), ..., R_n(h)$ が存在するとき, h の評価値 $Score_{lc+nt}$ を以下のように定義する。

$$Score_{lc+nt}(h) = \sum_{j=0}^{m}\sum_{k=0}^{n}\frac{N(L_j(h), Cat_h, R_k(h))}{N(L_j(h), R_k(h))}$$

3.3 仮説選択モジュールの処理

仮説選択モジュールは、前節の評価値を用いて仮説のスコアリングを行ない、上位のいくつかの仮説を、導入すべき文法の候補としてユーザーに提示する。

正解構文木集合からの局所文脈情報の取得

処理の前に、正解構文木集合に出現する各カテゴリの局所文脈情報を取得する必要がある。それからその頻度を調べ、データベースに格納しておく。また、文脈の出現頻度も調べてデータベースに格納する。

[1] lc : lexical category 語彙カテゴリ

[2] nt : nonterminal symbols 非終端記号

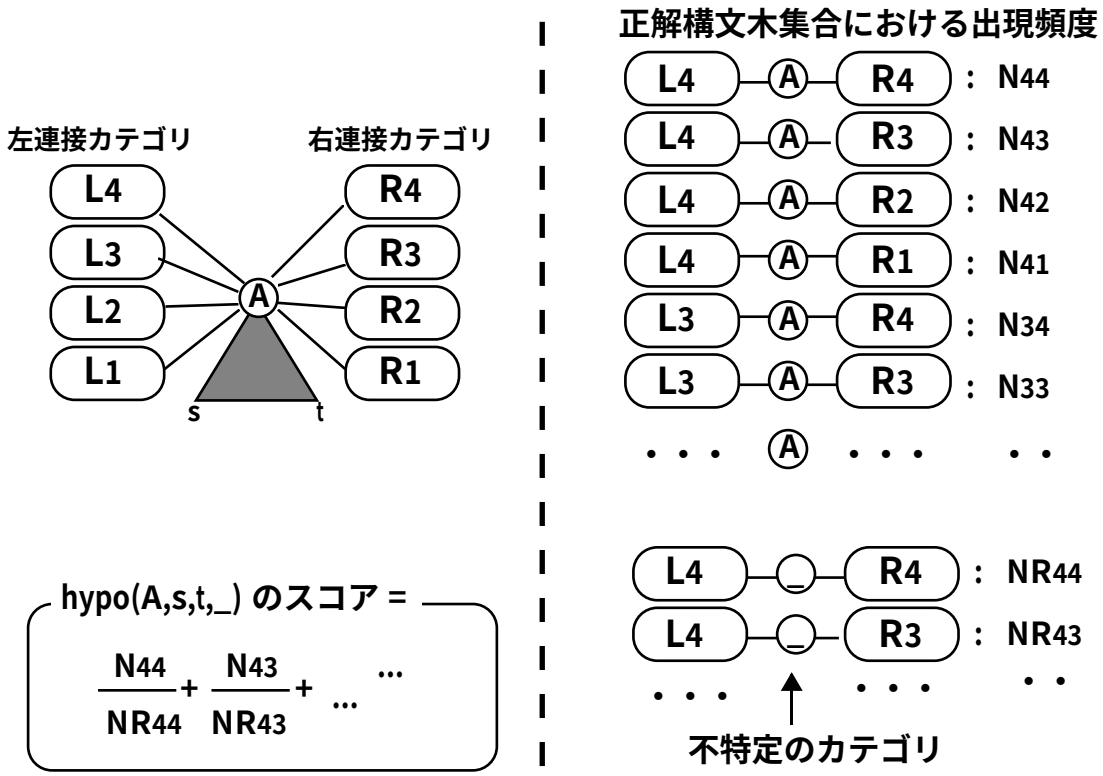

図 3.2: スコア計算

仮説スコアの算出

解析が失敗し、仮説が生成されたら、その仮説の左右の局所文脈情報を抽出する。局所文脈情報として語彙カテゴリと非終端記号の両方を考慮する場合、仮説の左右の不活性弧全てを環境とみなす。

得られた仮説の局所文脈情報それぞれに対して、仮説のラベルと文脈のセットの正解構文木集合中での出現頻度、同じく文脈の正解構文木中での出現頻度をデータベースから取り出し、スコアを算出する (図 3.2)。

仮説選択

仮説に付けられたスコアの上位いくつかの仮設をユーザーに提示する。

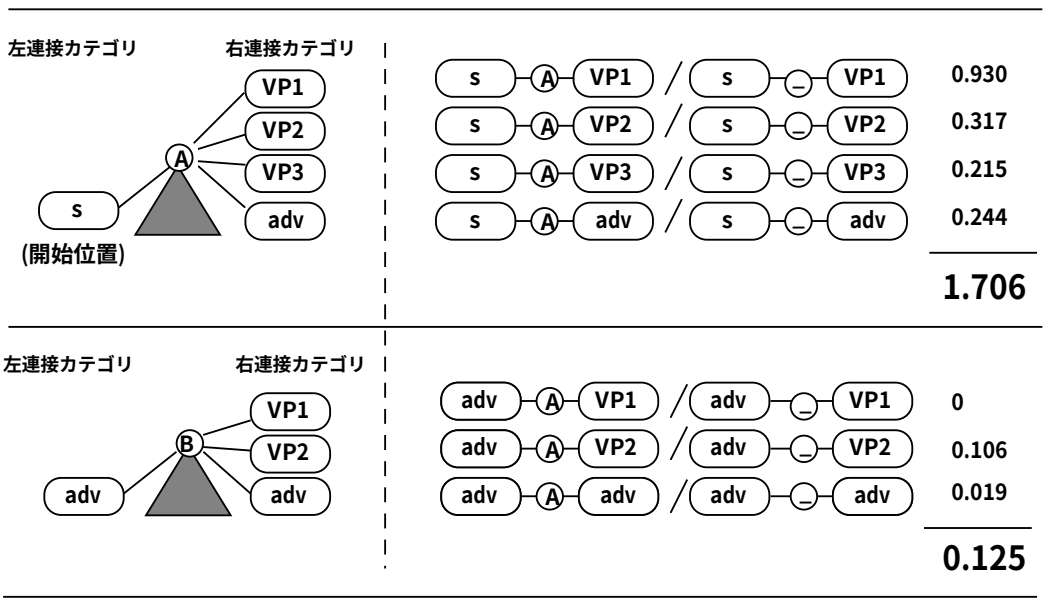

図 3.3: スコア計算の例

3.3.1 スコア計算の例

評価値 $Score_{lc+nt}$ におけるスコア計算の例を (図 3.3) に示す。この例では、共にいくつかの文脈を持つ仮説 A と仮説 B のスコアを算出した結果、仮説 A の方がより尤もらしいスコアだということになる。

第 4 章

実験と評価

この章では、前章で述べた方法を用いて実際に仮説生成、仮説選択を行なった結果を示し、評価と考察を行なう。

4.1 実験に使用した文法とコーパス

4.1.1 文法

実験で**初期文法**として用いた文法は、タナラックらの研究 [1] の結果得られた文法である。これは、EDR 英語コーパス [9] の品詞および括弧情報をもとに自動獲得された文法であり、非終端記号にはラベルとして機械的に割り振られた記号がつけられている (表 4.1 参照)。文法形式は文脈自由文法 (CFG) であり、規則数は 272 で、この中に語彙規則は含まれない。各規則の右辺 (RHS) には最大 4 つの語彙カテゴリまたは非終端記号が存在する (表 4.2参照)。

初期文法に含まれている非終端記号 (表 4.1における`l1n1,l1n2` など) は、コーパスから自動獲得する過程で機械的に割り当てられたものであり、55 種類ある。これらは名詞句や動詞句などの一般的なカテゴリと一致しない可能性がある。

初期文法に含まれる語彙カテゴリは、EDR 英語コーパス [9] に準ずるが、BLNK(空白文字) などいくつかの品詞は取り除かれている。語彙カテゴリ数は 18 で、実験で使用するコーパス内に出現する語彙カテゴリは全て含まれる (表 4.3参照)。

表 4.1: 初期文法の例

```
l1n1 -> adv,noun   l1n2 -> adj,noun   l1n3 -> adv,l1n3   ...
l1n1 -> adv,lin1   l1n2 -> adj,l1n1   l1n3 -> aux,vt     ...
l1n1 -> adv,l1n2   l1n2 -> adj,l1n2   l1n3 -> aux,l1n13  ...
l1n1 -> art,noun   l1n2 -> adj,l1n8   l1n3 -> l1n12,vt   ...
 ...        ...       ...        ...       ...       ...
```

表 4.2: 初期文法

規則数	272
非終端記号数	55
終端記号 (語彙カテゴリ) 数	18
右辺の項数の最大	4
右辺の項数の平均	2.04

表 4.3: 初期文法に含まれる語彙カテゴリのリスト

NOUN	名詞	VT	他動詞	AUX	助動詞
PRON	代名詞	BE	BE 動詞	PREP	前置詞
DEMO	指示詞	ADJ	形容詞	CONJ	接続詞
INDEF	不定語	ADV	副詞	ITJ	間投詞
WH	WH–語	PTCL	副詞小辞	UNIT	単位
VI	自動詞	ART	冠詞	NUM	数字

表 4.4: コーパスの例

```
((num,"1")((aux,"have")((pron,"you")((vt,"read")(((indef,"some")(noun,"p
art"))(((prep,"of")((adj,"each")((adj,"specific")(noun,"issue")))))((prep,
"of")((art,"the")(noun,"magazine")))))))))

(((art,"a")((adj,"Japanese")(noun,"girl")))((wh,"who")(((be,"is")(vt,"tra
in"))((((ptcl,"to")(vt,"provid"))(((adj,"entertaining") (conj,"and")(adj,
"lighthearted"))(noun,"company")))((adv,"especially")((prep,"for")(((art,
"a")(noun,"man"))(conj,"or")(((art,"a")(noun,"group"))((prep,"of")(noun,
"men")))))))))
```

表 4.5: コーパスのデータ

文数	48100
平均単語数	10.76
最大単語数	30

4.1.2 コーパス

実験に用いたコーパスは、EDR 英語コーパス [9] から品詞情報 (語彙カテゴリ) と括弧情報を抜きだしたものである。語彙カテゴリ以外の素性、例えば人称や数、に関する情報に関しては、本研究の対象外であるので考慮していない。実験では、品詞情報のみを用いて構文解析や文法仮説の生成を行ない、括弧情報は正解括弧付けとして、生成された構文木の正しさの評価に用いた。コーパスの平均単語数と最大単語数を表 4.5に、またコーパスのエントリの例を表 4.4に示す。

コーパスに対する文法の性能を知るために、あらかじめ**初期文法**を用いて構文解析を行なった。その結果が表 4.6 である。コーパスの正解括弧付けと交差しない[1]ものを正解とすると、60.7%のカバレッジを持つことがわかった。また、正解となった構文木のトップノードの分布から、仮説生成で使用するトップノードを l1n21 に決定した。

[1] 非交差の条件については次節で定義する

表 4.6: コーパスに対する初期文法のカバレッジ

コーパス全体 48100 文	括弧と交差しない構文木を生成 29195 文	トップノードが l1n21 17052 文
		トップノードが l1n21 以外 12143 文
	非交差構文木を生成できない 18905 文	構文木を生成 13822 文
		解析失敗 5083 文

4.2 仮説の正解を判定する評価基準

4.2.1 正解の基準

生成される仮説に対して、正解かどうか判定をする方法として、単純には人間が直接仮説を見て、自らの言語直観と照らしあわせ、受理できるかどうか判断することが考えられる。しかし、

- 規則に付けられたラベルは機械的な記号であるので、規則の右辺 (ラベル) と左辺 (子カテゴリ) の結び付きが正しいかどうか判別することは難しい。
- 仮説のラベルと子カテゴリだけを見て、正解と判断したとしても、文全体と文法全体から見て、その仮説から正しい構文木を生成できるかどうか判別するのは難しい。

ということから、この評価方法を実行することは難しい。

そこで本実験では、コーパスの正解括弧付けと非交差の構文木を生成し得るかどうかを正解の基準として用いる。

4.2.2 括弧付け非交差の判定

括弧付けは、括弧の位置を表す番号対の集合で表現することができる。

```
((pron,"he")((vt,"loves")((art,"a")(noun,"woman"))))
 0          1             2          3              4
```

各々の単語の間に付けられた番号が括弧位置である。この場合、単語noun,"woman"のみを覆う括弧は、$(3, 4)$ という番号対で表す。この文に付けられた全ての括弧を番号対で表

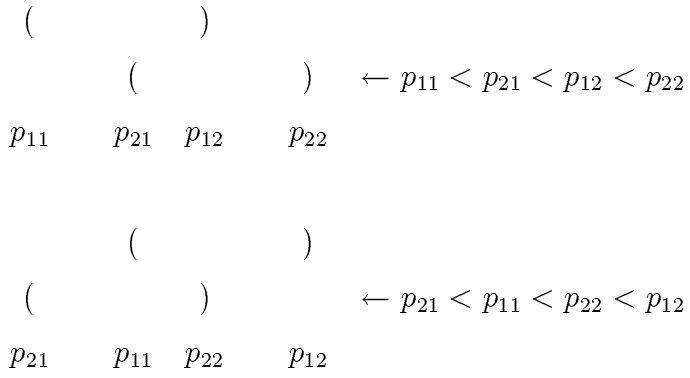

図 4.1: 括弧が交差する場合

すと、

$$Brc = \{(0,1),(1,2),(2,3),(3,4),(2,4),(1,4),(0,4)\}$$

となる。この集合と文の構文構造は 1 対 1 に対応する。

次に、この集合表現を用いて、括弧付けの非交差を定義する。

> 括弧集合 Brc_1 と Brc_2 において、括弧付けが非交差であるとは、
> $(p_{11}, p_{12}) \in Brc_1, (p_{21}, p_{22}) \in Brc_2$ なるすべての $p_{11}, p_{12}, p_{21}, p_{22}$ に対して、
> $p_{11} < p_{21} < p_{12} < p_{22}$ または $p_{21} < p_{11} < p_{22} < p_{12}$ という関係 (図 4.1)
> が成立しない場合を指す。

4.2.3 正解仮説の判定

仮説が正解であるということを、前項で定義した括弧付け非交差の条件を用いた以下の条件を満たす場合であると定義する。

- 子カテゴリ以下の括弧が全て非交差である。ただし、子カテゴリ以下の構造に曖昧性があり、括弧付けが複数ある場合は、ひとつでも非交差であればよい。
- その仮説と規則を用いて、括弧付けが非交差である構文木を組み上げることが可能である。ただし、構文木のトップノードカテゴリは仮説生成に用いたトップノードカテゴリと一致しなければならない。

表 4.7: 取得されたトライグラムに関するデータ

トライグラムの総数	3406741
トライグラムの異なり数	8051
環境 (左右の連接カテゴリの対) の異なり数	1562

ただし、現在の評価値はいずれも不活性弧仮説の子カテゴリに関する評価値を導入していないため、本実験では、同じ位置で同じラベルを持つ仮説をまとめて 1 つの仮説とみなし、仮説内部に非交差構文木がひとつでもあれば正解とした.

4.3 正解構文木集合の取得

仮説にスコアリングを行なう準備として、正解構文木集合に基づく品詞、非終端記号のトライグラムを取得する必要がある。

使用したコーパスは、品詞、括弧付きコーパスであり、非終端記号は付けられていないので、まず全ての例文を**初期文法**を用いてパースし、得られた構文木の中でコーパスにつけられた括弧と交差しない構文木を正解とし、それぞれにおいてトライグラムを取得した。括弧に非交差な構文木を生成した文の数は 29195 文で、そのなかに正解構文木の数は 33403 あった (表 4.6)。これは、正解の基準を括弧付け非交差としたために、一文において複数の構文木か正解とされている場合があるからである (一文に複数の構文木がある場合には、その文から得られたトライグラムの頻度が重複してカウントされるので、構文木の曖昧性の数で割った)。

取得されたトライグラムのデータを表 4.7に示す。

4.4 仮説生成モジュールでの失敗

初期文法で解析に失敗した 5082 文 (表 4.6の解析失敗の欄) を実験コーパスとして、仮説生成、仮説選択の実験を行なった。表 4.8にその結果を示す。

コーパス全体の 5082 文のうち、仮説生成モジュールが 1 つ以上の仮説を生成したものが 4730 文、残り 352 文は仮説が出力されなかった。また、仮説が生成されたものの、仮説

表 4.8: 仮説生成における失敗

実験コーパス 5082 文		
仮説生成に成功 3217 文	仮説生成で失敗 1865 文	
仮説が生成された 4730 文		失敗 352 文
仮説に正解 (非交差) が含まれた 3217 文	不正解 1513 文	

の中に正解が含まれなかった文が 1513 文あった。あわせて 1865 文は、仮説生成モジュールにおいて失敗となった文である。

4.5 仮説選択とその評価

前節の実験コーパスのうち、正解の仮説を生成できなかった文は除き、仮説生成で正しい仮説を生成することができた 3217 文に対して、仮説選択結果の評価を行なった。

4.5.1 結果と考察

表 4.9は、評価値 $Score_{lc+nt}$を用いた場合に、スコアの高いものから上位いくつかの仮説をユーザーに提示したとき、そこに正解が含まれるかどうかを文単位で数えたものである。提示する仮説数を上位 1 つだけにした場合、提示した仮説の中に正解を含んでいた文が 1362 あり、提示する仮説数を上位 5 つに拡大するとさらに 875 文で正解が含まれるようになり、上位 10 個に拡大するとさらに 397 文で正解が含まれるようになる、ということをこの表は示している。

評価値 $Score_{lc+nt}$を使用することによって、1 位の仮説のみを提示した場合、3217 文中で 1362 文において正解を提示することができた。上位 10 個の仮説を提示すると $1362 + 875 + 397 = 2634$ 文に対して正解を提示することができた。これは正解仮説が生成できた文のうちの 81.9%をカバーする。

表 4.10は、一文に対して提示する最大の仮説数を決めたときに、提示された仮説の中にどの程度正解が含まれているかを表している。提示する仮説数を 1 に絞った場合は、$\frac{1362}{3217} = 42.3\%$の確率で正解が提示され、提示する仮説数を 10 にした場合提示された仮説のうち $\frac{1362+3349+3409}{3217+11288+12846} \times 100 = \frac{8120}{27351} = 29.7\%$ が正解であった。

表 4.9: 正解を提示できた文の数 ($Score_{lc+nt}$)

提示仮説数	正解を提示できた文の数 (A)	$\frac{(A)}{\text{全文}}$
1	1362	42.3%
5	875	27.1%
10	397	11.7%
20	279	8.6%
30	98	3.0%
50	91	2.8%
51-	114	3.5%
全体	3217	(100%)

$Score_{lc}$と $Score_{lc+nt}$の比較

表 4.11と表 4.9 表 4.10と表 4.12 をそれぞれ見比べると、$lex + nonterm$ のほうが lex よりも、やや良い結果になっている。しかし、その差はそれほど大きくない。

$Score_{lc}$と $Score_{lc+nt}$で、それほど性能に差がつかなかったのは、$Score_{lc+nt}$のスコアにノイズが入っているためではないかと考えられる。すなわち、1. 仮説の左右に接するカテゴリは、全てがその仮説と共存するとは限らない。2. 複数存在する文脈はそれぞれ独立ではない、という問題があり、これがスコアに関してノイズとなっている可能性がある。

表 4.10: 提示された仮説の正解率 ($Score_{lc+nt}$)

提示仮説数	正解仮説数	仮説数	$\frac{\text{正解仮説数}}{\text{仮説数}}$
1	1362	3217	42.3%
5	3349	11288	29.8%
10	3409	12846	24.7%
20	4469	22105	19.5%
30	2606	17743	15.0%
50	2832	27001	11.2%
51–	6176	102015	6.2%
全体	24203	196214	13.3%

表 4.11: 正解を提示できた文の数 ($Score_{lc}$)

提示仮説数	正解を提示できた文の数 (A)	$\frac{(A)}{\text{全文}}$
1	1340	41.6%
5	1006	31.2%
10	277	8.6%
20	225	7.0%
30	111	3.5%
50	121	3.8%
51-	136	4.2%
全体	3217	(100%)

表 4.12: 提示された仮説の正解率 ($Score_{lc}$)

提示仮説数	正解仮説数	仮説数	$\frac{\text{正解仮説数}}{\text{仮説数}}$
1	1340	3217	41.6%
5	3368	11288	29.8%
10	3134	12846	24.3%
20	4300	22105	19.4%
30	2673	17743	15.0%
50	3033	27001	11.2%
51–	6315	102015	6.2%
全体	24203	196214	12.3%

第 5 章

結論

5.1 まとめ

本論文では、既存の文法に対し、特定のコーパスに対する適用範囲を拡大するために追加すべき規則の仮説を生成、選択する方法について述べた。なかでも、既存の文法で解析可能な文の構文木の情報を用いて、仮説選択のためのスコアリングを行なう方法を提案した。この手法は、ユーザーが指定する任意の一文に対して、現在の文法では解析できない文であるならば、文法規則の欠落を一つだけ補う規則の仮説を生成し、現在の文法で解析可能な文の構文木の集合から得られるトライグラムを用いた評価値を用いて仮説の順位付けを行ない、上位の仮説をユーザーに提示する。

実験において、上位 10 仮説に絞った場合には初期文法で解析できなかった文のうち、81.4%に対して正解を含む仮説を提示できることが確認された。上位 1 仮説に絞った場合には、解析できない文のうち 42.3%に対して正解仮説を提示することができた。

5.2 今後の課題

今後の研究課題として、以下のような事項が挙げられる。

- 子カテゴリに関するスコアの考慮
  現在採用しているスコアでは、同じ文の同じ位置にある同じラベルをもつインスタンス仮説はすべて同じスコアになる。これは外部環境のみのスコアを用いているた

めである。同じ文の同じ位置にある同じラベルをもつインスタンス仮説間の順位付けを行なうためには、何らかの内部的な評価値を導入する必要がある。

- 非終端記号にラベルの付いた構文付きコーパスにおける実験
  実験に用いたコーパスは、括弧情報は付いているが、非終端記号のカテゴリラベルはついていないので、生成された仮説の正解の基準として括弧の非交差を用いたが、これだと、適切な部分にに適切なカテゴリラベルを割り当てることができたかどうかの判定ができない。より厳密な判定をするために、非終端記号にラベルの付いた構文付きコーパスを使用して実験を行なう必要がある。

- 一般仮説の仮説選択、提示
  現在の実装では、一文に関するインスタンス仮説をユーザーに提示するモデルになっている。しかし、各文に対して仮説生成モジュールが必ず正解を含む仮説を生成できるわけではない。そこで、複数の文、あるいはコーパス全体に対して尤もらしい一般仮説を提示し、ユーザーに提示する方法を検討する必要がある。

- 尤もらしい構文木を提示
  また、現在の実装では、一文に関するインスタンス仮説そのものをユーザーに提示するが、その仮説を元に組み上げられるであろう構文木を作成して提示した方が、人間にとって正否の判断がつけやすいものと考えられる。ユーザにどのような情報を与えたら、ユーザーにとって使いやすい文法開発支援になるのか、さらに検討の余地があると思われる。

謝辞

本研究を進めるにあたり、終始御指導下さいました奥村学助教授に心から御礼申し上げます。

さらに、本研究に関し助言を頂きましたタナラック・ティラマヌコン助手に感謝致します。

また、本研究の初期段階において、ツールを御提供いただきました松下電器産業株式会社マルチメディアシステム研究所の清野正樹氏に感謝致します。

最後に、本研究に関して熱心な意見、討論を交わして頂いた奥村研究室の皆様に感謝します。

参考文献

[1] ティラマヌコン・タナラック, 奥村学, 括弧付きコーパスを利用した文法獲得手法, 情報処理学会研究報告, NL–116–13, pp85–92, 1996.

[2] Kiyono, M.,Tsujii, J., Linguistic Knowledge Aqcuisition from Parsing Failures, Proc. of 6th Conference of the European Chapter of the Association for Computational Linguistics (EACL '93) pp222–231, 1993.

[3] Kiyono, M.,Tsujii, J., Combination of Symbolic and Statistical Approaches for Grammatical Knowledge Acquisition, Proc. of 4th Conference on Applied Natural Language Processing, ACL, pp72–77, 1994.

[4] Kiyono, M.,Tsujii, J., Hypothesis Selection in Grammar Acquisition, Proc. of 15th International Conference on Computational Linguistics (COLING '94), pp837–841, 1994.

[5] 大谷耕嗣, 中川聖一, CFG とバイグラムの結合による文法の半自動修正法, 情報処理学会研究報告, 95–SLP–9, pp99–104, 1995.

[6] Ootani, K., Nakagawa, S., A Semi-Automatic Learning Method of Grammar Rules for Spontaneous Speech Proc. of Natural Language Processing Pacific Rim Symposium '95 (NLPRS '95) pp514–519, 1995.

[7] Brill, E., Marcus, M., Automatically acquiring phrase structure using distributional analysis, Proc. of Speech and Natural Language Workshop, pp155–159, 1992.

[8] 森信介, 長尾真, 統計によるタグ付きコーパスからの統語規則の獲得, 情報処理学会研究報告, 95–NL–110, pp79–86, 1995.

[9] (株) 日本電子化辞書研究所, EDR 電子化辞書仕様説明書 (第 2 版), 1995.

[10] 白井 清昭, 徳永健伸, 田中穂積, 括弧付きコーパスからの日本語確率文脈自由文法の自動抽出 自然言語処理, Vol. 4, No. 1, pp125–146, 1996.

[11] 今井宏樹, 一般化 LR 法を用いた文中の任意の構文的誤りの検出と訂正 修士論文 北陸先端科学技術大学院大学情報科学研究科 情報処理学専攻, 1996.

[12] 宇津呂武仁, 松本裕治, コーパスを用いた言語知識の獲得, 人工知能学会誌, Vol. 10, No. 2, pp197–204, 1995.

[13] 松本裕治, 頑健な自然言語処理へのアプローチ, 情報処理学会学会誌, Vol. 33, No. 7, pp757–767, 1992.

[14] 辻井潤一, 視点の変換, 人工知能学会誌, Vol. 11, No. 4, pp530–541, 1996.