T2R2 東京科学大学 リサーチリポジトリ
Science Tokyo Research Repository

論文 / 著書情報
Article / Book Information

題目(和文)	統計情報を利用した統合的自然言語解析
Title(English)	
著者(和文)	白井清昭
Author(English)	Kiyoaki Shirai
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第3875号, 授与年月日:1998年3月26日, 学位の種別:課程博士, 審査員:
Citation(English)	Degree:Doctor of Engineering, Conferring organization: Tokyo Institute of Technology, Report number:甲第3875号, Conferred date:1998/3/26, Degree Type:Course doctor, Examiner:
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Powered by T2R2 (Science Tokyo Research Repository)

統計情報を利用した統合的自然言語解析

白井 清昭
東京工業大学大学院 情報理工学研究科 計算工学専攻

目 次

第 1 章　序論 **1**

1.1　背景と目的 1

1.2　本論文の構成 4

第 2 章　関連研究 **5**

2.1　統合処理における種々の統計情報の取り扱い 5

2.2　品詞連接に関する優先度，単語の出現頻度 7

2.3　構文的優先度 8

2.4　語彙的従属関係 9

2.5　構文的制約と語彙的従属関係の統合 10

2.6　距離に関する優先度 14

2.7　本章のまとめ 15

第 3 章　種々の統計情報を統合した日本語文解析 **18**

3.1　統合的確率言語モデル 18

3.1.1　構文モデル $P(R)$ 20

3.1.2　語彙モデル $P(W|R)$ 22

3.1.2.1　単語生成文脈 22

3.1.2.2　単語生成文脈決定規則 23

3.1.2.3　従属係数 29

3.1.3　語義モデル $P(S|W,R)$ 32

3.1.4　本節のまとめ 33

3.2　評価実験 34

3.2.1　構文モデルの学習 35

3.2.1.1　文節ラベル 36

3.2.1.2　文節の係り受け解析を行う CFG 38

3.2.1.3　構文モデルのパラメタ推定 39

3.2.2　語彙モデルの学習 39

3.2.2.1　格要素に関する従属係数の学習 40

3.2.2.2　用言に係る助詞に関する従属係数の学習 42

3.2.2.3　体言に係る助詞に関する従属係数の学習 45

3.2.2.4　格要素以外の名詞の従属係数の学習 45

3.2.3　実験結果 45

3.2.4　考察 51

3.2.4.1　並列構造 52

3.2.4.2　補助用言 53

3.2.4.3 実験で考慮しなかった語彙的従属関係 55

第 4 章 最大エントロピー法の効率化 59

4.1 最大エントロピー法 60
4.2 GIS アルゴリズムの効率化 64
4.3 素性選択アルゴリズムの効率化 66
4.4 素性効用に基づく素性選択 68
4.4.1 $U_T(f)$ の計算方法 70
4.4.2 $U_H(f)$ の計算方法 73
4.5 評価実験 75
4.5.1 素性候補の集合 S の作成 76
4.5.2 確率モデル推定 78

第 5 章 文脈自由文法の自動獲得 81

5.1 文法獲得に関する既存研究 81
5.1.1 平文コーパスからの文法獲得 82
5.1.2 品詞タグ付きコーパスからの文法獲得 83
5.1.3 括弧付きコーパスからの文法獲得 83
5.1.4 構文木付きコーパスからの文法獲得 85
5.1.5 既存手法の問題点 86
5.2 CFG 獲得アルゴリズム 87
5.2.1 EDR コーパスの概要 87
5.2.2 構文構造の内部ノードへの非終端記号の付与 87
5.3 文法の改良 90
5.3.1 不正な規則の検出・削除 91
5.3.2 文法サイズの縮小 93
5.3.3 文法が生成する解析木数の抑制 95
5.3.3.1 同一品詞列の取り扱い 95
5.3.3.2 品詞の細分化 96
5.3.3.3 法・様相を表わす助動詞に対する構造の統一 97
5.4 評価実験 98
5.4.1 文法獲得 99
5.4.2 構文解析 99

第 6 章 結論 103

謝辞 105

参考文献 106

付 録 A 目標事象を二分したときのエントロピーの計算 114

第1章　序論

1.1　背景と目的

情報化・国際化が飛躍的に進む今日において，情報検索や機械翻訳など，自然言語を計算機によって処理する自然言語処理の重要性が見直されている．特に，自然言語で表現された文書や音声を計算機で処理するためには，その内容を理解する自然言語解析は必須の技術であり，日夜研究が進められている．自然言語解析は，具体的には以下の 4 つの処理を行うことと仮定されるのが一般的である．

1. 形態素解析
   入力文に含まれる単語の品詞を解析・決定する．日本語，韓国語，タイ語などの単語の区切りが明示されていない言語については，単語区切りを認定することも形態素解析に含まれる．
2. 構文解析
   単語の係り受け関係の決定や句の範囲の認定など，入力文の構文構造を解析する．
3. 意味解析
   入力文が持つ意味を解析・決定する．具体的には，各単語の意味 (語義) を決定する語義曖昧性解消や，動作主格・目的格などといった動詞と名詞の間の関係を表わす深層格の解析などが挙げられる．
4. 文脈・談話解析
   1 つの文を解析の対象とするのではなく，文の集合である文章・談話がどのような意味を持つかを解析する．具体的には，文間の関係を決定する談話構造の解析や，指示詞が具体的に何を指すのかを決定する照応解析などが挙げられる．

これらの解析を行う際には 2 つのアプローチがある．1 つはこれらの解析を逐次的に行う方法，もう 1 つはこれらの解析を統合的に行う方法である．逐次処理とは，上に挙げた 4 つの解析をこの順序で段階的に行い，かつそれぞれの解析を単独で行うことを指す．すなわち，最初に形態素解析を行った後，その出力をもとに構文解析を行う．同様に構文解析の出力をもとに意味解析を行い，最後に意味解析の出力をもとに文脈・談話解析を行う．ところが，自然言語処理におけるこれらの問題は単独に解決できるわけではなく，お互いに関連している．例えば，形態素解析を単語の形態的な情報のみを用いて行うのは不可能であり，入力文全体に関する構文的な情報や意味的な情報を使わなければ正解を得られない場合がある [1]．そのため，例えば構文的な情報を用いれば誤りであると判断できる解析結果の候補も，形態的な情報のみを用いて行う形態素解析の段階では出力される．逐次処理においては，このような無駄に生成される解析結果の候補に対しても次段以降の処理を行わなければならないために，多くの解析を段階的に行えば行うほど処理しなければならない解析結果の候補の数も組み合わせ的に増大するという問題点がある．これに対して，自然言語処理における複数の解析を同時に行う統合処理においては，形態的な情報，構文的な情報，意味的な情報などを同時に用いて自然言語解析を行うため，逐次処理のように無駄な解析結果の候補を出力することはない．したがって，自然言語を解析する際には複数の解析を統合的に処理することが望ましい．

自然言語を解析するためには，逐次処理を行う場合も統合処理を行う場合も，自然言語処理用知識と呼ばれる知識が必要となる．例えば，形態素解析では単語の形態的な情報 (品詞など) を記載した辞書などが，構文解析では文法などが，意味解析では動詞が取り得る格およびその格要素としてどのような名詞が現わ

れ得るかを記述した格フレームなどが，自然言語処理用知識として用いられる．ところが，自然言語において観察される現象は多様であり，これら全ての言語現象を網羅的に解析するための自然言語処理用知識を作成するのは容易ではない．また，自然言語を解析する際には，最終的に得られる解析結果が一意に決まることは稀であり，一般に複数の解析結果の候補が出力される．したがって，これら複数の解析結果の候補の中から正しい解を選択しなければならない．これは一般に曖昧性解消と呼ばれ，自然言語処理における中心的な課題である．

近年，自然言語処理におけるこのような問題を解決する方法として，統計的自然言語処理と呼ばれる研究が注目を浴びている．統計的自然言語処理とは，自然言語処理における様々な問題の解決に何らかの統計情報を利用する研究であり，次の二種類に大きく分類される．

- スコア付けによる曖昧性解消に関する研究
- 自然言語処理用知識の自動獲得に関する研究

統計的自然言語処理が盛んに行われるようになった要因としては，文書の電子化がすすみ統計情報の学習に利用可能なコーパス (新聞，雑誌など日常で使われている自然言語文を収集した例文データベース) が大量に普及し始めたことや，辞書やシソーラス (単語の意味的な分類を記述した知識ベース) など機械可読な言語資源の整備が進んだことが挙げられる．

統計的自然言語処理の中で最も盛んに行われているのが，コーパスなどの言語資源から得られる統計情報を利用して解析結果の候補に対して何らかのスコア付けを行い，最もスコアの高い解析結果を正しい解とみなして出力することにより曖昧性を解消する研究である．ところが，前述したように自然言語解析は統合処理によって行うことが理想的だが，統合処理の枠組において統計情報を利用して曖昧性を解消するには以下のような問題がある．

- 統合処理では様々なタイプの統計情報を取り扱わなければならない．少なくとも，形態素解析・構文解析などの個々の解析に有効な統計情報は利用するべきである．ところが，このような様々なタイプの統計情報を同時に学習することはほとんど不可能である．なぜなら，解析結果の候補に与えるスコアに反映させる統計情報の種類を増やせば増やすほど，そのスコアのパラメタの学習に要するデータ量も爆発的に増大するからである．したがって，現在利用できる言語資源から学習可能な統計情報を個別に学習し，それらを最終的に何らかの形で組み合わせることによって，様々なタイプの統計情報を全体のスコアに反映させなければならない．

- 統合処理によって曖昧性解消を行うために，既存の研究によって提案・学習された統計情報をそのまま組み合わせることは一般に困難である．なぜなら，統計情報を利用した過去の研究においては，逐次処理を行うことを前提に，個々の解析を単独で行うことを目的としたものが多く，また個々の解析に特化したスコア付けを行っているため，これを直接組み合わせることはできないからである．例えば，英語の前置詞句の係り先を決定する PP-attachment 問題を統計的手法を用いて解決する研究がいくつかあるが [26, 77]，これらの研究によって提案されたスコアは PP-attachment 問題に特化している．そのため，PP-attachment 問題も含めた様々な解析を同時に取り扱う統合処理において，その解析結果の候補に与えるスコアに PP-attachment 問題を解くことのみを考慮したスコアをどのように組み込めばよいのかについては自明ではない．したがって，個々の統計情報を学習する際には，それをスコア全体にどのように反映させるかを十分考慮に入れる必要がある．

- 過去の研究においては，統計情報を利用したスコア付けを行う際に，そのスコアの値自身が持つ意味が明確にされていない場合がある．しかしながら，曖昧性解消に用いるスコアの意味を明確にすることは，統合処理を行う際には重要である．なぜなら，スコアに様々な種類の統計情報が反映されてい

る場合，そのスコアを利用した曖昧性解消がうまくいかなかった場合には，解析に失敗した原因としてどの種類の統計情報が不適切であるのかを判断しなければならない．この際，個々のスコアの持つ意味を明確にすることは，そのスコアが適切に学習されているのか否かの判断を容易にし，解析に失敗した原因を調査する手助けとなるからである．

本研究では，以上の問題点を考慮し，統計情報を曖昧性解消に用いる新しい枠組として統合的確率言語モデルを提案する [32, 33, 34, 93, 94, 97]．この統合的確率言語モデルの特徴を以下にまとめる．

◇ 以下に挙げる統計情報を統合的に取り扱う．

- 主に形態素解析の曖昧性解消に有効に働く統計情報
  品詞連接に関する優先度，単語の出現頻度
- 主に構文解析の曖昧性解消に有効に働く統計情報
  構文的優先度，距離に関する優先度，語彙的従属関係
- 主に語義曖昧性解消に有効に働く統計情報
  語義の出現頻度，語義の共起関係

◇ 曖昧性解消に用いるスコアの意味付けとして確率理論を用いる．

◇ 個々の統計情報の学習を確率理論に基づいて個別に行う．また，個別に学習された確率 (統計情報) の積を解析結果の候補に与えるスコアとすることにより，個別に学習された統計情報を同時に利用して曖昧性を解消する．

スコア付けによる曖昧性解消を目的とした統計的自然言語処理においてもうひとつ大きな問題となっているのは，スコアを学習するためのデータ量が十分にないという，いわゆるデータスパースネス問題である．特に，曖昧性解消に有効な統計情報の中でも，語彙的従属関係を学習する際にはこのデータスパースネス問題が発生しやすい．語彙的従属関係とは，例えば「食べる」という動詞が「りんご」などの食物を表わす名詞を目的格として取りやすいなどの，単語の共起関係に関連した統計情報全般を指す．構文解析の精度を向上させるには，このような語彙的従属関係をスコアに反映させる必要があることは以前から指摘され，実際に語彙的従属関係をスコアに取り入れる様々な試みが行われている [2, 14, 16, 17, 23, 25, 28, 51, 56, 82, 101]．しかしながら，単語の数は膨大であるため，単語の共起に関する統計情報である語彙的従属関係を学習するためには，現時点で利用できる言語資源だけでは学習に必要なデータ量を十分に確保することができない．このようなデータスパースネス問題に対応するために，数多くのスムージング手法が提案されてきた．特に，スムージング手法のひとつとして近年注目を集めているのが最大エントロピー法による確率モデル推定である．最大エントロピー法は，確率モデルを推定する際に様々な素性を同時に取り扱うことができるという利点を持つ．また素性によって示唆される事象の集合に重なりがあってもよく，それだけ複雑な確率モデルの推定にも応用できる．自然言語において観察される事象はいろいろな要因 (素性) が複雑に絡み合っていると考えられるので，このような特性を持つ最大エントロピー法は自然言語処理に適したスムージング手法であるといえる．実際，最大エントロピー法を自然言語処理に応用した研究もいくつか報告されている [5, 20, 72, 73, 78, 81, 104]．しかしながら，この最大エントロピー法は確率モデルの推定に要する計算量が多く，語彙的従属関係のようなパラメタ数の多い確率モデルの推定に適用することができないといった問題点があった．そこで本研究では，最大エントロピー法による確率モデルの推定を効率良く行うための手法を提案する [92, 96]．

自然言語を解析する際には，以上で述べたような曖昧性解消問題の他にも，自然言語処理に用いる知識の不備により解析に失敗するといった問題もある．例えば，辞書に登録されていない単語 (未知語) を入力に

含む文を解析することはできない．しかしながら，自然言語において用いられる単語の全てを人手によって辞書に登録することはほとんど不可能である．このような問題を解決する手段の 1 つとして，コーパスなどの言語資源から自然言語処理用知識を自動的に獲得することが挙げられる．言語資源から自動的に獲得された知識は，人手によって記述された知識に比べて，知識作成に要する人的負担が軽い，得られた知識が作成者の主観に影響されにくいといった利点を持つ．また，自然言語処理用知識を人手によって作成する場合には作成者が予期しなかった言語現象に関する情報が欠落する可能性があるが，知識源として大量の言語資源を用いれば多くの言語現象に関する情報を得ることができる．すなわち，自動獲得された知識は人手によって作成された知識に比べて適用範囲が広いという特長がある．

言語資源から自然言語処理用知識を自動的に獲得する研究としては，格フレームを自動獲得する研究 [27] やシソーラスを自動構築する研究 [29] などがある．文法もまた自動的に獲得できる自然言語処理用知識の 1 つである．文法とは構文解析を行うための知識であり，その文法クラスとしては主に文脈自由文法 (Context-Free Grammar，以下 CFG) が用いられる．また CFG を利用した構文解析のための効率的なアルゴリズムとして一般化 LR 法 [103] (Generalized LR Method，以下 GLR 法)，チャート法などがある．そのため，CFG および CFG を拡張した確率文脈自由文法 (Probabilistic Context-Free Grammar, 以下 PCFG) を自動獲得する手法がいくつか提案されている [39, 48, 62, 79, 108]．本研究でも CFG を自動獲得する一手法を提案する [88, 89, 90, 91, 95]．本研究で提案する手法の特長は，人間が持っている言語学的知識を使わずに言語資源のみを利用して CFG を獲得しようとする過去の研究のアプローチとは異なり，言語資源と言語学的知識の両方を利用して CFG を獲得することにより，CFG 獲得に要する計算量を大幅に削減したことにある．

本研究の目的を以下にまとめる．

- 自然言語解析を統合的に処理することを前提に，その曖昧性を解消するために，複数の統計情報を統合的に取り扱う統合的確率言語モデルを提案する．
- パラメタ数の多い確率モデルの推定にも応用できるように，最大エントロピー法による確率モデル推定アルゴリズムの効率化を行う．
- 言語資源と言語学的知識の両方を利用して効率良く CFG を自動獲得する手法を提案する．

1.2 本論文の構成

本論文の構成を以下に述べる．

2 章では，まず統合処理における曖昧性の解消に複数の統計情報を同時に利用する際に注意しなければならない点について述べる．次に統計情報を利用して曖昧性解消を行う過去の研究とその問題点について概説する．

3 章では，複数の統計情報を取り扱うことに重点を置いた統合的確率言語モデルについて詳説する．また，この統合的確率言語モデルをコーパスから学習し，それを用いて日本語文の係り受け解析実験を行うことにより，本研究で提案した統合的確率言語モデルの評価を行う．

4 章では，まず最大エントロピー法による確率モデル推定アルゴリズムを概説し，次にそれを効率化するいくつかの手法を提案する．また，提案した手法の評価実験についても述べる．

5 章では，括弧付きコーパスから CFG を自動獲得する手法を提案し，さらに獲得した CFG を洗練するためのいくつかの手法について説明する．また自動獲得した CFG の評価実験についても述べる．

最後に 6 章において本研究を総括し，今後の課題について述べる．

第2章　関連研究

本章では，統計的自然言語処理の中でも，曖昧性解消を目的とした統計情報の学習に関する過去の研究について述べる．まず 2.1 節では，統合処理における曖昧性の解消に複数の統計情報を利用する際に注意しなければならない点について述べる．2.2〜2.6 節では曖昧性解消に有効であると考えられる具体的な統計情報について述べ，またそれらの統計情報を曖昧性解消に利用した過去の研究例について述べる．最後に 2.7 節では，本章のまとめとしてこれら過去の研究を総括する．

2.1　統合処理における種々の統計情報の取り扱い

1.1 節で述べたように，自然言語解析における形態素解析，構文解析，意味解析，文脈・談話解析といった様々な処理は統合的に行うことが望ましい．ところが，統合処理を行う際には出力される解析結果の候補の数も多くなり，これらから正しい解を選択する曖昧性解消もそれだけ難しくなる．

統合処理によって曖昧性を解消する際にも，統計情報を用いた解析結果の候補に対するスコア付けは有効である．ここで注意しなければならないのは，統合処理では複数の統計情報を同時に取り扱う必要があるということである．例えば，形態素解析と構文解析を同時に行う場合には，形態素解析に有効な統計情報と構文解析に有効な統計情報を解析結果の候補に与えるスコアに反映させなければならない．このように複数の統計情報を同時に取り扱う際には，以下の 4 つの条件を満たすことが重要である．

条件 1　曖昧性解消に有効な複数の統計情報が解析結果の候補に与えられるスコアに反映されていること

形態素解析，構文解析などを統合的に処理し，その曖昧性を統計情報を利用して解消するためには，個々の処理の曖昧性解消に有効な全ての統計情報が解析結果の候補に与えられるスコアに反映されるべきである．

条件 2　解析結果に与えられるスコアが個々の統計情報を反映するスコアから構成的に計算できること

例えば，解析結果の候補 I に対するスコアを $S(I)$ とし，これに n 種類の統計情報を反映させる場合には，$S(I)$ がそれぞれの統計情報のみを反映したスコア $S_i(I)$ から構成的に計算できることが望ましい．

$$S(I) = f(\ S_1(I), S_2(I), \cdots, S_{n-1}(I), S_n(I)\ ) \tag{2.1}$$

式 (2.1) における f は，$S_i(I)$ から $S(I)$ を計算するための関数を表わす．ここで重要なのは，どのような関数を f とするかということではなく，最終的なスコア $S(I)$ が個々の統計情報を反映したスコア $S_i(I)$ の組に分解できるという点である．この理由を以下に述べる．

- 統計情報の学習が容易である

  式 (2.1) のようにスコア全体を $S_i(I)$ の組として分解できなくても，n 種類の統計情報を同時に反映させたスコア $S_{1,2,\cdots,n}(I)$ を利用できれば，同程度の精度で曖昧性を解消することができる．

しかしながら，多くの統計情報を反映させればさせるほど，スコア $S_{1,2,\cdots,n}(I)$ を与えるモデルの推定パラメタ数も多くなり，それだけ大量の学習データを必要とする．それに加えて，複数の統計情報を同時に反映したモデルを学習するためには，モデルに反映するべき全ての統計情報を取り出すための知識が学習用データとして用いる言語資源に付加されていなければならない．ところが，このような言語資源は一般に作成コストが高く，大量に用意することは難しい．これに対し，$S(I)$ を式 (2.1) のように $S_i(I)$ の関数として分解できれば，$S_i(I)$ を個別に学習することができる．$S_i(I)$ には一種類の統計情報のみしか反映されていないので，比較的少ないデータ量で学習することができる．

統計情報を個別に学習するもうひとつの理由として，統計情報の種類によって学習に要する言語資源の質・量が異なることが挙げられる．例えば，統計情報のひとつである構文的優先度を学習する場合には，学習用言語資源として構文構造が付加されたコーパス (以下，構文構造付きコーパス) が必要となる[1]．この構文構造付きコーパスは比較的作成コストが高いが，構文的優先度の推定パラメタ数はそれほど多くはないので，比較的少ないデータ量で学習することができる．これに対して，語彙的従属関係 (単語の共起関係) は，構文構造付きコーパスに比べて作成コストの低い品詞タグ付きコーパス (単語の品詞のみが付加されたコーパス) を用いても十分学習することが可能である．しかしながら，一般に語彙的従属関係の推定パラメタは非常に多く，学習に大量のデータを必要とする．このように，統計情報の種類によって学習に要する言語資源の質・量が異なるので，個々の統計情報を個別に学習し，それらを組み合わせることによって，最終的なスコアに種々の統計情報を反映させることが望ましい．

- 曖昧性解消時における個々の統計情報の役割の理解が容易である

  不適切な統計情報をスコアに反映させたり，あるいは統計情報の学習がうまくいかなかった場合には，スコア $S(I)$ を用いた曖昧性解消に失敗することが考えられる．ところが，統合処理における曖昧性解消のためのスコア $S(I)$ には様々な種類の統計情報が反映されている．したがって，曖昧性解消に失敗した原因を調べる際に，全ての統計情報を同時に学習したスコア $S_{1,2,\cdots,n}(I)$ を用いた場合には，どの統計情報が不適切であるのか，あるいはどの統計情報の学習がうまくいかなかったのかを判断することが難しい．これに対し，全体のスコアを個々の統計情報を反映したスコア $S_i(I)$ の関数として分解することができれば，曖昧性解消に失敗した原因の調査を容易に行うことができる．したがって，曖昧性解消時におけるスコア $S(I)$ の特性を調べる際には，式 (2.1) のように個々の統計情報を反映するスコア $S_i(I)$ を組み合わせて全体のスコアとするモデルが理想的である．

条件 3　個々の統計情報を反映させたスコアが確率的な意味を持っていること

統計情報を学習する際に，そのスコア $S_i(I)$ の値が確率的な意味を持つことは以下の 2 つの点で重要である．

- $S_i(I)$ の持つ確率的な意味が不明確である場合には，それらをどのように組み合わせればよいのか，すなわち式 (2.1) における関数 f として何を用いればよいのかについては自明ではない．例えば，スコア $S_1(I)$ が取り得る値の範囲が $[0, 100]$ で，$S_2(I)$ が取り得る値の範囲が $[-1, 1]$ である場合，両者を単純に足し合わせたとすれば，最終的なスコアには $S_2(I)$ によって学習された統計情報がほとんど反映されないのは明らかである．仮に両者に何らかの正規化を施したとしても，両者の重みは $1:1$ でよいのか，あるいはどちらか片方にいくらか重みをおくべきなのか

[1] Inside-Outside アルゴリズム [48] に代表されるような EM アルゴリズムを用いて，構文構造が付加されていないコーパスから構文的な統計情報を学習する研究も行われている．しかしながら，このような教師なしの学習は一般に精度が悪く，現時点では構文構造が付加されたコーパスを利用した方が品質の良い統計情報を学習できると考えられる．

については，$S_1(I)$ および $S_2(I)$ の値の持つ意味が明確でない限り判断が難しい．それぞれのスコアに対する重みを実験的に求める方法も考えられるが，組み合わせるスコアが複数ある場合を考えると現実的ではない．これに対し，個々のスコア $S_i(I)$ が確率的意味を持つように，すなわち $S_i(I)$ をある事象に対する確率として学習し，また解析結果の候補 I の生成確率を学習された個々の確率の積によって計算することができれば，その生成確率を I に与える最終的なスコアとすることにより，個々のスコア $S_i(I)$ を自然に組み合わせることができる．

- 曖昧性解消に失敗した原因を調査する際には，スコア $S_i(I)$ が確率的な意味を持つ方が，そのスコアが適切であるかどうかを判断しやすい．

条件 4　全体のスコアに組み込むことができる形で個々の統計情報を学習すること

たとえ個々のスコア $S_i(I)$ の持つ確率的な意味が明確であっても，それらが全体のスコア $S(I)$ に組み込むことができなければ意味がない．例えば，2 つのスコア $S_1(I)$ と $S_2(I)$ をお互いに矛盾するような仮定の下で学習した場合には，当然それらを組み合わせることはできない．統合処理によって曖昧性を解消する際には，逐次処理の場合と異なり，個々の統計情報を全体のスコアにどのように反映するかを考慮して統計情報を学習しなければならない．

次節以降では，過去の研究における統計情報の学習及びそれらを用いた曖昧性解消手法について述べる．

2.2　品詞連接に関する優先度，単語の出現頻度

形態素解析の曖昧性解消には品詞連接に関する優先度と呼ばれる統計情報がよく用いられる．品詞連接に関する優先度とは，例えば “動詞” という品詞を持つ単語の後には “語尾” という品詞を持つ単語が現われやすいなど，どのような品詞の並びが現われやすいかを示した統計情報を指す．この品詞連接に関する優先度を学習するモデルとしては，以下に示す品詞の n-gram モデルが挙げられる．

$$P(l_1,\cdots,l_m)=\prod_{i=1}^{m}P(l_i|l_{i-1},l_{i-2},\cdots,l_{i-n+1}) \tag{2.2}$$

$$\text{但し，}\quad \begin{array}{ll} P(l_i|l_{i-1},l_{i-2},\cdots,l_{i-n+1})=P(l_i|l_{i-1},l_{i-2},\cdots,l_1) & if\ \ 1<i<n \\ P(l_i|l_{i-1},l_{i-2},\cdots,l_{i-n+1})=P(l_1) & if\ \ i=1 \end{array}$$

品詞 n-gram は品詞列 $l_1,\cdots,l_m$ の生成確率を与える確率モデルである．式 (2.2) は，先頭から i 番目の品詞 l_i の生成確率がその直前の n 個の品詞のみに依存すると仮定し，品詞列 $l_1,\cdots,l_m$ の生成確率を $P(l_i|l_{i-1},l_{i-2},\cdots,l_{i-n+1})$ の積として計算することを意味する．実際には，$n=2$ の bi-gram モデル (式 (2.3))，$n=3$ の tri-gram モデル (式 (2.4)) がよく用いられる．

$$P(l_1,\cdots,l_m) = P(l_1)\cdot\prod_{i=2}^{m}P(l_i|l_{i-1}) \tag{2.3}$$

$$P(l_1,\cdots,l_m) = P(l_1)\cdot P(l_2|l_1)\cdot\prod_{i=3}^{m}P(l_i|l_{i-1},l_{i-2}) \tag{2.4}$$

品詞 n-gram が品詞連接に関する優先度のみを学習するための確率モデルであるのに対し，隠れマルコフモデル (Hidden Markov Model，以下 HMM) は品詞連接に関する優先度と単語の出現頻度を同時に考慮するモデルである [69]．単語の出現頻度は，例えば同じ “名詞” を品詞とする単語でも「私」はよく現われるが「拙者」はあまり現われないなど，ある品詞からどのような単語が生成されやすいかを示した統計情報で

ある．HMM の例を式 (2.5) に示す．

$$P(w_1, \cdots, w_m, l_1, \cdots, l_m) = \prod_{i=1}^{m} P(w_i|l_i)P(l_i|l_{i-1}, l_{i-2}) \tag{2.5}$$

HMM は品詞列 $l_1, \cdots, l_m$ と単語列 $w_1, \cdots, w_m$ の生成確率を与える確率モデルである．式 (2.5) において，$P(l_i|l_{i-1}, l_{i-2})$ は品詞 tri-gram であり，品詞連接に関する優先度が反映されている．一般には，tri-gram だけでなく任意の n-gram を用いることができる．また，$P(w_i|l_i)$ は品詞 l_i から単語 w_i が生成される確率を表わしており，単語の出現頻度が反映されている．この HMM によって計算される生成確率は，形態素解析結果の候補に対するスコアとして利用することができる．実際，HMM を用いた形態素解析は，英語 [13, 18, 44] においても日本語 [63, 65, 66] においても数多くの研究が報告されている．

2.3 構文的優先度

構文解析は CFG を利用して行うのが一般的である．CFG は以下に示すような書き換え規則の集合から成る．

$$A_i \rightarrow \zeta_i \tag{2.6}$$

この書き換え規則は，左辺の非終端記号 A_i は右辺の非終端記号または終端記号の列 ζ_i に書き換え可能であることを示す．例えば，規則 (2.7) は S(文) が PP(後置詞句) と VP(動詞句) に展開されることを表わし，規則 (2.8) は PP が N(名詞) と P(助詞) に展開されることを表わしている．

$$S \rightarrow PP \quad VP \tag{2.7}$$

$$PP \rightarrow N \quad P \tag{2.8}$$

このような書き替え規則を繰り返し適用することにより，開始記号 S から最終的に入力文の品詞列 $l_1, \cdots, l_n$ もしくは単語列 $w_1, \cdots, w_n$ が生成され，また適用された書き換え規則 r_i によりその構文構造が特定される．CFG によって特定された構文構造は一般に木構造によって表わされる．

構文的優先度とは，簡単に言えば入力文の構文構造としてどのような木構造が現われやすいかを表わした統計情報であり，構文解析の曖昧性解消に用いられる．この構文的優先度を学習するモデルとしては，CFG を拡張した PCFG が挙げられる．PCFG とは，各規則 $r_i : A_i \rightarrow \zeta_i$ に対して確率 $P(r_i)$ を付与した文法である．$P(r_i)$ は，左辺の非終端記号 A_i が与えられた時に，それが右辺の記号列 ζ_i に書き換えられる条件付き確率を表わす．したがって，$P(r_i)$ は任意の非終端記号 A に対して式 (2.9) の条件を満たさなければならない．

$$\sum_{\forall\, r_i\, :\, A \rightarrow \zeta_j} P(r_i) \;=\; 1 \tag{2.9}$$

構文構造の生成確率は，構文構造の生成に用いられた規則 r_i の確率 $P(r_i)$ の積として計算される．PCFG によって計算される生成確率は構文解析結果の候補に対するスコアとして利用することができる．

PCFG は文脈自由文法を基礎にしているため，文脈依存性を取り扱うことができない．しかしながら，完全な文脈依存文法 (Context-Sensitive Grammar，I 型文法) を構文解析に利用するのは，効率的な解析アルゴリズムが発見されていないために困難である．そこで，構文解析そのものは CFG を用いて行い，確率モデルに若干の文脈依存性を反映させる試みもいくつか行われている．

まず，構文解析の際に規則が適用される順序を考慮し，式 (2.10) に示すように，規則の適用確率の前件として既に適用された規則の集合を与える研究がいくつかなされている．

$$P(r_i|r_1, \cdots, r_{i-1}) \tag{2.10}$$

式 (2.10) において，$r_1, \cdots, r_{i-1}$ は r_i の前に適用された規則を表わす．以下，これを導出履歴と呼ぶ．式 (2.10) に示す確率モデルの推定パラメタ空間は膨大で，これを直接学習することは一般には不可能である．したがって，導出履歴を近似して縮退することにより，推定パラメタ空間を縮小しなければならない．この導出履歴を近似する方法として，例えば以下の 3 つが提案されている．

$$P(r_i|r_1, \cdots, r_{i-1}) = P(r_i|r_{i-1}) \tag{2.11}$$

$$P(r_i|r_1, \cdots, r_{i-1}) = P(r_i|E(r_1, \cdots, r_{i-1})) \tag{2.12}$$

$$P(r_i|r_1, \cdots, r_{i-1}) = P(r_i|r'_i) \tag{2.13}$$

式 (2.11) は，規則の適用確率は直前に用いられた規則 r_{i-1} のみに依存するとしたモデルである [38]．一方式 (2.12) は，導出履歴をグルーピングする関数 E を用いたモデルである [6]．Black らは，この関数 E を決定木によって学習している．また式 (2.13) は，適用する規則 $r_i : A_i \rightarrow \zeta_i$ の左辺の非終端記号 A_i を生成した規則を r'_i とし，規則の適用確率はこの r'_i のみに依存するとしたモデルである [15, 55]．いずれにせよ，導出履歴を確率モデルの前件に加えることにより，文脈依存性を取り扱うモデルとなっている．

一方 Sekine らは，CFG 規則で用いる非終端記号を S(文) と NP(名詞句) に限定することにより，非終端記号による構文情報の抽象化を制限し，CFG が持つ文脈自由性の問題を和らげる手法を提案している [86]．Sekine らは，S または NP を根ノードとする部分木を Penn Treebank[58] から自動的に抽出し，その部分木を，部分木の根ノード (S または NP) を左辺とし部分木の葉に相当する記号列 ζ_i を右辺とする CFG 規則に変換している．また，構文解析の際に用いられた CFG 規則から入力文の構文構造を決定する際には，各 CFG 規則に対応した部分木を組み合わせることにより全体の構文構造を決定している．しかしながら，この手法は深さ 2 以上の部分木に CFG 規則を対応させることを意味し，このため規則の数及び推定パラメタの数が爆発的に増大するといった問題点もある．

2.4 語彙的従属関係

語彙的従属関係とは一般に単語共起に関する統計情報を指す．例えば,「履く」という動詞はその目的語として「靴」や「ズボン」といった単語と共起しやすいが,「犬」と共起することは滅多にない．このとき,「履く」と「靴」には正の相関関係が,「履く」と「犬」には負の相関関係があると考えられる．語彙的従属関係は，このような動詞と格要素の間の従属関係だけではなく，動詞と助詞との間の従属関係，格間の従属関係など様々な種類がある．

語彙的従属関係を曖昧性解消に利用した例としては，名詞の共起情報を利用して，複合名詞を構成する名詞間の係り受け解析を行う小林らの研究が挙げられる [42, 43]．小林らは，2 つの名詞 n_1, n_2 の従属関係 (共起関係) を相互情報量 MI によって評価している．

$$MI(n_1, n_2) = \log \frac{C(n_1, n_2)}{C(n_1)C(n_2)} \tag{2.14}$$

式 (2.14) において，$C(n_i)$ は名詞 n_i の出現頻度を，$C(n_1, n_2)$ は名詞 n_1 が名詞 n_2 に係るという事例の出現頻度を表わす．$MI(n_1, n_2)$ は，n_1 と n_2 に正の相関関係があれば正の値を取り，負の相関関係があれば負の値を取る．また，両者に相関関係がなければ 0 に近い値を取る．そして，複合名詞の係り受け解析結果の各候補に対して，係り受け関係にある名詞 n_1, n_2 の相互情報量 $MI(n_1, n_2)$ の幾何平均をスコアとして与えることにより曖昧性解消を行っている．

複合名詞の係り受け解析だけではなく，PP-attachment 問題もまた語彙的従属関係が曖昧性解消に有効に働く問題のひとつである．PP-attachment 問題とは，単語列 “v, n_1, p, n_2” が与えられたとき (v は動詞,

n_i は名詞，p は前置詞を表わす)，“p, n_2” から構成される前置詞句 (Prepositional Phrase，PP) が v と n_1 のどちらに係るかを決定する問題である．例えば，以下の例文 (1) においては前置詞句 “in the morning” は動詞 “ate” に係り，例文 (2) においては前置詞句 “on the table” は名詞 “glasses” に係る．

(1) I ate breakfast in the morning.

(2) I broke glasses on the table.

前置詞句の係り先を決定するためには語彙的な情報が必要不可欠である．例えば，2 つの例文 (1),(2) の品詞列は同じなので，品詞に関する統計情報 (品詞連接に関する優先度など) のみを用いても前置詞句の正しい係り先を決定することはできない．Hindle は，式 (2.15) に示す LA(Lexical Association) と呼ばれる統計量を利用して PP-attachment 問題を解決する手法を提案している [26].

$$LA(v, n_1, p) = \log_2 \frac{P(verb_attach\ p|v, n_1)}{P(noun_attach\ p|v, n_1)} \tag{2.15}$$

式 (2.15) の分子は，動詞とその目的語 v, n_1 が文中に現われたときに，前置詞 p が動詞 v に係る確率である．同様に，式 (2.15) の分母は前置詞 p が名詞 n_1 に係る確率である．すなわち，$LA(v, n_1, p)$ は v, n_1, p, n_2 が与えられたときに，p が v に係る確率と n_1 に係る確率の比を表わしており，この値によって前置詞句の係り先を決定する．この統計量はコーパスから収集された PP-attachment 問題の正解事例から推定する．また Resnik は，この LA を拡張し，単語そのものを用いる代りに単語の意味クラスを用いた CA(Conceptual Association) という統計量により PP-attachment 問題を解決する手法を提案している [77]．そして，LA の推定に使用可能な学習事例の数が少ない場合に限って LA の代わりに CA を用いることにより，PP-attachment 問題の曖昧性解消の精度が向上したと報告している．

2.5 構文的制約と語彙的従属関係の統合

2.4 節で述べた研究は，語彙的従属関係のみを統計情報として用い，複合名詞の係り受け解析や PP-attachment 問題といった入力文全体ではなく一部分の係り受け解析を対象としたものである．これに対し，入力文全体の構文構造を決定する構文解析を対象に，構文的優先度と語彙的従属関係を組み合わせて曖昧性解消を行う研究もある．一般に，CFG を用いて構文解析を行う際には，品詞列を入力としてその構文構造を決定する場合が多いため，語彙的従属関係は無視される．しかしながら，先ほどの PP-attachment 問題と同様に，構文構造を決定する際には構文的な統計情報を考慮するだけでは不十分であり，単語の共起関係などの語彙的な統計情報が必要となる．したがって，構文的優先度と語彙的従属関係を組み合わせることによって構文解析の解析精度を向上させる試みは近年積極的に行われている．本節では，このような語彙的従属関係を利用して構文解析を行う過去の研究について概観する．

構文解析の曖昧性解消に語彙的従属関係を利用する方法としては，まず PCFG を拡張する方法が挙げられる．Hogenout らは，PCFG における書き換え規則の非終端記号に，その非終端記号が支配する句の主辞となる単語を付加することによって，PCFG による確率モデルに語彙的従属関係を反映させる方法を提案している [28]．例えば，動詞 “eat” とその目的格となる名詞 “apple” の語彙的従属関係は以下のような規則によって確率モデルに反映される．

$$VP \quad \rightarrow \quad V{:}eat \quad NP{:}apple \tag{2.16}$$

以降では，このように非終端記号に主辞となる単語を付加することを PCFG の語彙化と呼ぶ．Charniak もまた PCFG を語彙化することにより語彙的従属関係を解析結果の候補に与えるスコアに反映させてい

る [14]．Charniak は，Hogenout らと同様に非終端記号に主辞となる単語を付加しているが，規則の確率を計算する際に，主辞となる単語の生成確率 (式 (2.16) の例では “eat” が生成される確率) と主辞となる単語を無視した規則の確率 (式 (2.16) の例では規則 “VP → V NP” が適用される確率) とに分けて計算している．また Collins らは，PCFG の語彙化に加えて，下位範疇化情報を確率モデルに組み込む方法を提案している [17]．下位範疇化情報とは，語彙的知識のひとつであり，個々の動詞が取る主格や目的格などの必須格，もしくは修飾句などに関する情報を指す．しかしながら，PCFG の語彙化によって構文的優先度と語彙的従属関係を組み合わせる方法は，非終端記号に単語を付加することによって規則の数が組み合わせ的に増大するため，推定するパラメタ数も非常に多くなるといった問題がある．これに対して Hogenout らや Charniak，Collins は，パラメタ推定の際に様々なスムージング方法を適用している．また田辺らは，CFG 規則の数を減らすために，非終端記号に主辞となる単語そのものを付与する代わりに，単語の意味クラスを付与する試みを行っている [101]．PCFG を語彙化する際のもうひとつの問題は，構文的優先度と語彙的従属関係を同時に学習しなればならないという点にある．これは学習用データを十分に得ることができない可能性があるだけではなく，学習した確率モデルの特性の分析を難しくする要因にもなる．

Li らは，構文解析結果の候補 I に対して，構文的優先度を反映させた確率モデル $P_{syn}(I)$ と語彙的従属関係を反映させた確率モデル $P_{lex3}(I)$，$P_{lex2}(I)$ を個別に学習するモデルを提案している [50, 51]．

$$P_{lex3}(I) = \left(\prod_{i}^{m} P(n|v,s)\right)^{1/m} \tag{2.17}$$

$$P_{lex2}(I) = \left(\prod_{i}^{m} P(s|v)\right)^{1/m} \tag{2.18}$$

$$P_{syn}(I) = \left(\prod_{i}^{m} P(d \rightarrow d_1, \cdots, d_n|r_i)\right)^{1/m} \tag{2.19}$$

式 (2.17) における $P(n|v,s)$ は，動詞 v の格スロット s に名詞 n が現われる確率を表わしている．例えば，構文解析結果の候補 I において，動詞 eat の格スロット $with$ に名詞 $spoon$ が現われたとき，$spoon$ の生成確率を以下のように計算する．

$$P(spoon \mid eat, with) \tag{2.20}$$

$P_{lex3}(I)$ は，I に含まれる全ての格フレームに現われる名詞 n の生成確率の幾何平均である．生成確率の幾何平均をスコアとしたのは，I によって積を取る確率の数 (式 (2.17) における m) が異なる場合を考慮したためである．同様に，$P_{lex2}(I)$ は動詞 v がスロット s を生成する確率 $P(s|v)$ の幾何平均である．例えば，I において動詞 eat がスロット $with$ を取るとき，以下の確率が $P_{lex2}(I)$ の要素として計算される．

$$P(with \mid eat) \tag{2.21}$$

これに対し，式 (2.19) における $P(d \rightarrow d_1, \cdots, d_n|r_i)$ は距離確率と呼ばれるものであり，係り受け関係にある単語間の距離を反映したものである．この距離確率の例を以下にあげる．

$$P(8 \rightarrow 2, 6 \mid NP \rightarrow NP\ PP) \tag{2.22}$$

$$P(8 \rightarrow 5, 3 \mid NP \rightarrow NP\ PP) \tag{2.23}$$

式 (2.22) は “$NP \rightarrow NP\ PP$” という規則が適用されたときに，規則の右辺に位置する NP の支配する単語の数が 2，PP の支配する単語の数が 6 となる確率を表わし，式 (2.23) は NP の支配する単語の数が 5，PP の支配する単語の数が 3 となる確率を表わしている．式 (2.22) は PP の主辞が NP の主辞に距離 2 で係ることを示し，式 (2.23) は距離 5 で係ることを示す．一般に，係り受け関係にある 2 単語間の距離が近

い解釈が正しい解釈として優先される傾向があるので，式 (2.22) は式 (2.23) よりも高い値を持つと予想される．$P_{syn}(I)$ は，I を生成した全ての CFG 規則の距離確率 $P(d_1, \cdots, d_n|r_i)$ の幾何平均であり，係り受け関係にある単語間の距離の平均を構文的優先度とみなして学習している．Li らの手法では，語彙的従属関係は構文的優先度に優先するといった心理言語学原理に基づき，まず P_{lex3} を I に与えるスコアとして曖昧性解消を行う．そして，曖昧性解消に失敗したとき (1 位の解と 2 位の解のスコアの差が十分に大きくなかったとき) には，I のスコアとして P_{lex2}, P_{syn} を順番に与えている．したがって，構文的優先度と語彙的従属関係をそれぞれ独立に学習しているものの，これらを同時に利用して曖昧性解消を行うわけではない．もちろん，$P_{syn}(I)$, $P_{lex2}(I)$, $P_{lex3}(I)$ を何らかの形で組み合わせて I のスコアとすれば，構文的優先度と語彙的従属関係を同時に考慮することも可能である．しかしながら，それぞれのスコアは確率の幾何平均であるために，これらの値が持つ確率的な意味が明確ではなく，3 つのスコアをどのようにして組み合わせれば良いのか (例えば，各スコアの重みをどのように決定すれば良いのか) は自明ではない．実際，Li らは $P_{syn}(I) \times P_{lex3}(I)$ を I のスコアとして実験したところ曖昧性解消の精度が悪くなったと報告しているが，これは $P_{syn}(I)$ と $P_{lex3}(I)$ の重みを 1 : 1 としたことが適切ではなかったことが原因のひとつとして考えられる．

Jelinek，Magerman らは，構文的優先度と語彙的従属関係の両方を用いて構文解析を行う SPATTER パーザと呼ばれる構文解析器を提案している [37, 56, 57]．SPATTER パーザは，ラベリングステップと拡張ステップの 2 つのステップによって構文解析を行う．ラベリングステップでは，木構造の内部ノードの非終端記号もしくは単語の品詞を決定し，またその確率を以下のように計算する．

$$P(N_l^k \mid context) \tag{2.24}$$

SPATTER パーザにおいては，非終端記号も品詞もラベル l として等しく取り扱われ，式 (2.24) における N_l^k は，内部ノードまたは単語 N^k のラベル (非終端記号または品詞) が l であることを表わしている．式 (2.24) における $context$ は，N^k の左右に隣接するノード，さらにその左右に隣接するノード，N^k の子ノードの列のうち最も左にある 2 つのノード，最も右にある 2 つのノードなどの集合である．各ノードにはそのノードの主辞となる単語が付加されているため，語彙的な情報も式 (2.24) の確率の前件に含まれる．一方，拡張ステップは自分自身と親ノードとの関係を決定し，またその確率を以下のように計算する．

$$P(N_e^k \mid context) \tag{2.25}$$

e の値は，N^k がその親ノードの一番左の子ノードに相当する場合には “right”，一番右の子に相当する場合には “left”，一番左でも右でもない場合には “up”，親ノードの唯一の子である場合には “unary” のいずれかの値を取る．このステップは構文構造を決定することに相当する．また，式 (2.25) における $context$ は式 (2.24) と同様に定義される．この $context$ には様々な素性が含まれているために非常に多様であり，式 (2.24),(2.25) の確率を直接推定することは不可能である．そこで彼らは，$context$ に含まれるどの素性が曖昧性解消に有効であるかを確率決定木を用いて自動的に学習している．このように自動学習した SPATTER パーザは，人手で構築したパーザよりも解析精度が向上したと報告している．しかしながら，SPATTER パーザは PCFG の語彙化と同様に構文的優先度と語彙的従属関係を同時に取り扱うモデルとなっているため，2.1 節で述べた条件 **2** を満たしていない．

PCFG を基礎にした構文解析では，入力文の構文構造は木構造で表現される．これに対し，入力文の構文構造を依存構造 (入力文中の各単語の係り先のみを特定した構造) で表現し，依存構造に対してスコア付けを行うことにより曖昧性を解消する手法もいくつか提案されている．Alves は日本語を対象に，文節間の係り受け関係を依存構造で表わし，依存関係にある単語対 (w_i, w_j) の相互情報量の積を依存構造 D のスコアとする方法を提案している [2]．

$$Score(D) = \prod MI(w_i, w_j) \tag{2.26}$$

例えば，ある依存構造 D において，文節「靴が」が文節「履く」に係っているときには，相互情報量 MI(靴, 履く) を式 (2.14) により計算し，これを $Score(D)$ の要素とする．これにより,「靴」と「履く」がどれだけ共起しやすいかといった語彙的従属関係をスコアに反映させることができる．しかしながら，Alves のモデルは語彙的従属関係のみを考慮したものであり，構文的優先度は全く考慮されていない．

Collins は英語を対象に，依存構造に対してその生成確率を与えるモデルを提案している [16]．まず，入力文 S が与えられたとき，これを base NP(主辞として 1 つの名詞のみを含むような最小の名詞句) と他の構成素 (動詞など) とに分割し，これら構成素の集合 B を認定する．B の要素は日本語における文節に相当すると考えられる．次に，$B = \{b_1, \cdots, b_n\}$ の間に存在する依存関係の集合を D とし，式 (2.27) に示す確率をもとに D の生成確率を計算する．

$$P(R \mid \langle w_i, t_i \rangle, \langle w_j, t_j \rangle) \tag{2.27}$$

式 (2.27) は，2 つの単語 w_i, w_j が同じ文中に現われたとき，それらに R という統語的関係が存在する確率を示している．ここで，$\langle w_i, t_i \rangle$ は B 中の i 番目の構成素 b_i の主辞となる単語および品詞であり，R は単語 w_i と w_j の統語的関係の種類 (例えば，w_i と w_j は動詞とその主語という関係にある，など) を表わしている．式 (2.27) は，ある単語の組に対して，それらの間にどのような統語的関係が存在するのか，もしくは統語的関係は存在しないのかを予測するモデルとなっているので，構文的優先度と語彙的従属関係の両方を確率モデルに反映させることができる．

藤尾らは，Collins らと同様の手法を用いて日本語を対象とした文節の係り受け解析を行う方法を提案している [24, 25]．藤尾らは，入力文における文節間の係り受け関係が互いに交差しないという制約の下で，それぞれの文節の係り先を決定している．このとき，2 つの文節間に係り受け関係が存在する確率を以下のように計算する．

$$P(R, \langle w_j, t_j \rangle \mid \langle w_i, t_i \rangle) \tag{2.28}$$

式 (2.28) は，ある単語 w_i が存在したときに，それが別の単語 w_j に関係 R で係る確率を表わす．単語 w_i の係り先を決定する際に，係り先の単語 w_j によって確率モデルの前件が変化する式 (2.27) と異なり，式 (2.28) では確率モデルの前件は常に等しいので，同じ信頼度を持つ確率モデル (推定に用いられた学習データ量が等しい確率モデル) の比較によって w_i の係り先を決定できる．但し，藤尾らは文節の係り先を決定しているので，式 (2.28) における $\langle w_i, t_i \rangle$ は実際には文節 f_i となる．また，文節 f_i を表わす素性として，その文節において最も主要な意味を持つ単語，文節の係り属性・受け属性，文節内の読点の有無などを用いている．

Eisner もまた依存構造に対してその生成確率を与える確率モデルを提案している [22, 23]．Collins や藤尾らは形態素解析の結果，すなわち単語列とその品詞列を入力として依存構造の生成確率を計算するのに対し，Eisner は品詞付けと依存構造の決定を同時に，すなわち形態素解析と構文解析を同時に行う．また，Collins のように入力文を基本構成素 B に分割してから構成素間の依存関係を解析するのではなく，入力文中に含まれる全ての単語の依存関係を解析する．Eisner は，単語列を W, 品詞列を L, 依存関係の集合を D として，これらの生成確率 $P(W, L, D)$ を解析結果の候補に対するスコアとして曖昧性を解消している．

$$P(W, L, D) = P(W, L) \times P(D|W, L) \tag{2.29}$$

ここで，$P(W, L)$ は単語列・品詞列の生成確率であり，式 (2.5) に示した HMM を用いて推定される．これに対して $P(D|W, L)$ は依存構造 D の生成確率であり，式 (2.27) に示すような単語 w_i と w_j の間に依存関係が存在する確率をもとに計算される．このモデルを用いた解析実験の結果，解析精度は Collins のモデル [16] とほぼ同等であった．また，形態素解析と構文解析 (この場合は依存構造の決定) を同時に行うことにより，形態素解析を単独で行うときよりも解析精度が向上したと報告している．

依存構造をベースにしたこれらの手法は，基本的に係り受け関係にある二単語の従属関係しか考慮しない．例えば,「彼女が先生になる」という文の依存構造においては,「彼女」と「先生」はともに動詞「なる」

に係り，両者の間の依存関係は明示されない．しかしながら，例えば「机が先生になる」という文は通常非文となるように，同じ動詞「なる」のガ格とヲ格の格要素となっている「彼女」と「先生」の間にも依存関係 (これを格要素間の従属関係という) は存在する．このように，依存構造を基盤にした手法は，依存構造に明示されない格要素間の従属関係などの語彙的従属関係を取り扱いにくいという欠点がある．

Schabes は CFG に代わる構文解析手段として SLTAG(Stochastic Lexicalized Tree Adjoining Grammar) を提案した [76, 82]．TAG(Tree Adjoinning Grammar) は文法の基本構成要素として初期木 (initial tree) と補助木 (auxiliary tree) という 2 種類の木構造を用意し，代入 (substitution) と接続 (adjoining) という 2 つの操作を用いてこれらの木構造を組み合わせ，入力文全体の構文構造を完成させることにより構文解析を行う枠組である [105]．TAG は木構造を基本要素としているため，CFG では取り扱いにくかった長距離依存性を自然に記述できるという利点がある．LTAG(Lexicalized TAG) は，TAG に対して，初期木と補助木の葉に少なくとも 1 つの語彙項目 (単語) が存在しなければならないという制約を付け加えたものである [83, 85]．基本構成要素に語彙項目を必ず付加させることにより，語彙的従属関係と構文的優先度を自然に組み合わせることができる．SLTAG(Stochastic LTAG) は，代入または接続という操作に PCFG と同様の確率を割り当てるように LTAG を拡張したものである．代入または接続という操作の確率の積を構文解析結果の候補のスコアとすることにより，語彙的従属関係を曖昧性解消に利用することができる．しかしながら，先ほどの PCFG を語彙化するアプローチと同様に，基本要素となる初期木と補助木が構文構造と単語の組み合わせとなるため，これらの数が爆発的に増大するといった問題がある．

2.6 距離に関する優先度

日本語では，入力文中の各単語は位置的に近い単語に係る傾向がある．例えば Maruyama らは，文節間の係り受け関係においては，位置的に近い文節間の係り受け関係は遠いものよりも高い頻度で発生すると報告している [59]．また，英語では，構成要素は遠くの句構造よりも近くの句構造に取り込まれやすいといった right association と呼ばれる優先規則が古くから提唱されている [64]．このような係り受け関係にある単語間の距離に関する統計情報を距離に関する優先度と呼ぶ．

距離に関する優先度を曖昧性解消に利用する試みもいくつか行われている．Li らが提案した距離確率 (式 (2.19)) は距離に関する優先度を学習していると考えられる [50, 51]．また，これと同様の試みは Hogenout らによっても行われている [28]．Hogenout らは，CFG 規則の非終端記号に，その非終端記号が支配する単語の数 m を付加することにより，PCFG による確率モデルに距離に関する優先度を反映させている．例えば，$A \rightarrow B_1\ B_2$ という規則は，B_1, B_2 が支配する単語の数 n_1, n_2 によって，以下のように細分化される．

$$A \quad \rightarrow \quad B_1{:}n_1 \quad B_2{:}n_2 \tag{2.30}$$

規則 (2.30) は，B_2 の主辞となる単語が B_1 の主辞となる単語に距離 n_1 で係ることを意味する．したがって，n_1 の値が大きくなればなるほど，その規則の適用確率は小さくなると予想される．

Li らや Hogenout らのモデルは PCFG を基礎とし，これを距離に関する優先度を取り扱えるように拡張したものである．ところが，このように PCFG を拡張したモデルは，PCFG を語彙化したときと同様に，非終端記号がシンボルとそれが支配する単語数 n の組み合わせとなるために規則数が組み合わせ的に増大する．さらに，PCFG を拡張したモデルは，近い要素に係りやすいといった距離に関する優先度が正しく反映されないこともある．例えば，4 つの名詞 N からなる複合名詞の構造として，図 2.1 の (a),(b) を考える．

図 2.1 の (a),(b) は，左から 2 番目の名詞の係り先のみ異なる．したがって，近い単語に係る (a) の構造により高い確率を与えるはずである．ところが，PCFG を拡張した Li らのモデルにおいては，この 2 つの構造の生成確率は以下のように計算される．

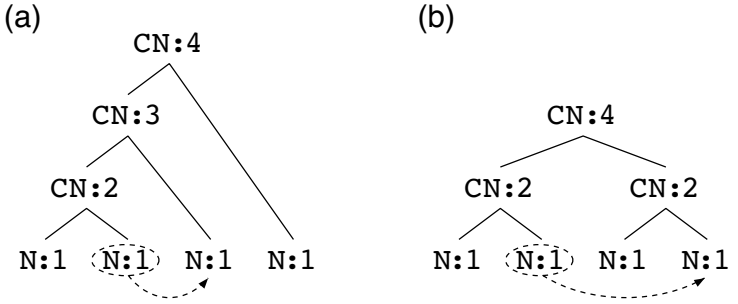


図 2.1: 距離に関する優先度を反映するために拡張された PCFG

$$(a) \quad P(4 \rightarrow 3\ 1 \mid CN \rightarrow CN\ N) \cdot P(3 \rightarrow 2\ 1 \mid CN \rightarrow CN\ N) \tag{2.31}$$

$$(b) \quad P(4 \rightarrow 2\ 2 \mid CN \rightarrow CN\ CN) \tag{2.32}$$

これらの式において，$P(2 \rightarrow 1\ 1 \mid CN \rightarrow N\ N)$ の値は 1 であるので省略している．式 (2.31) の値は 2 つの確率の積となっているため，式 (2.32) の値よりも小さくなると考えられる．これは，近い単語に係る (a) の構造に低い確率を与えることになり，距離に関する優先度が PCFG を拡張したモデルに必ずしも正しく反映されないことを示唆する．

これに対し，依存構造の生成確率を計算する Collins [16]，藤尾ら [24, 25]，Eisner [22, 23] のモデルにおいても，式 (2.27) を以下のように拡張することにより，距離に関する優先度をモデル全体に反映させている．

$$P(R \mid \langle w_i, t_i \rangle, \langle w_j, t_j \rangle, dist(i,j)) \tag{2.33}$$

式 (2.33) において，$dist(i,j)$ は単語 w_i, w_j 間の距離を表わす．すなわち，式 (2.33) は，2 つの単語 w_i, w_j が同じ文中に距離 $dist(i,j)$ で現われたとき，それらに R という依存関係が存在する確率を表わす．したがって，$dist(i,j)$ の値が小さければ小さいほど式 (2.33) の値も大きくなると予想される．

2.7 本章のまとめ

本章で述べたように，自然言語処理における様々な種類の問題に対して，統計情報を利用して曖昧性解消を行う研究が行われ，その成果が報告されている．しかしながら，表 2.1 に示すように，2.1 節で述べた条件 **1**〜条件 **4** を同時に満たす手法は存在しない．

まず，過去に提案された手法の中には，逐次処理を行うことを前提に，特定の処理に特化した研究が多く見られる．例えば，2.2 節で挙げた研究は形態素解析のみを，2.3 節で挙げた研究は構文解析のみを，2.4 節で挙げた研究は複合名詞の係り受け解析や PP-attachment 問題のみをそれぞれ対象とし，条件 **1** を満たしていない．しかも，これらの手法の中には，特定の処理の曖昧性解消を行うためだけに統計情報を学習し，他の統計情報と組み合わせて用いることを考慮していない，すなわち条件 **4** を満たしていないものもある．例えば，形態素解析の曖昧性解消に用いられる品詞の n-gram モデルは品詞列 L の生成確率 $P(L)$ を計算し，構文解析の曖昧性解消に用いられる PCFG は構文構造 R の生成確率 $P(R)$ を計算する．ところが，構文構造 R には品詞列も含まれているので，PCFG は品詞列 L を生成する確率モデルでもある．したがって，形態素解析と構文解析を統合的に処理して曖昧性解消を行う際に，表 2.1 の条件 **4** の欄の “$*$” で示し

表 2.1: 既存研究のまとめ

既存研究		条件 **1**	条件 **2**	条件 **3**	条件 **4**
品詞連接 [13, 18, 44, 63, 65, 66]	(2.2 節)	×			*
構文的優先度 [38, 6, 15, 55, 86]	(2.3 節)	×			*
語彙的従属関係 [42, 43, 26, 77]	(2.4 節)	×			×
構文的優先度+語彙的従属関係 [28, 14, 17, 101, 37, 56, 57, 22, 23]	(2.5 節)	○	×	○ 式 (2.34)	
構文的優先度+語彙的従属関係 [16, 24, 25]	(2.5 節)	○	×	○ 式 (2.35)	
距離 [50, 51]	(2.5 節)	○	○	×	

た 2 つの統計情報，すなわち品詞 n-gram と PCFG によって与えられる生成確率を組み合わせたスコアを利用しようとする場合，両者はともに品詞列 L を生成する確率モデルとなっているので，どちらにどれだけ重みをおけばよいのかなど，その最適な組み合わせ方は自明ではない．また，2.4 節で述べた LA [26] や CA [77] などの統計量は，PP-attachment 問題を解くことのみに特化しているため，やはり他の統計情報と組み合わせる際に問題が生じる．

一方，2.5 節で述べた手法は，構文的優先度と語彙的従属関係の両方を学習するものであり，構文解析と形態素解析を統合的に処理した研究もある．しかしながら，これらの研究の多くは構文的優先度と語彙的従属関係を同時に学習するモデルとなっているため，条件 **2** を満たしていない．また，Li らが提案した手法 [50, 51] は，構文的優先度と語彙的従属関係を個別に学習するモデルとなっているが，個々のスコアが持つ確率的な意味が明確でない，すなわち条件 **3** を満たしていないために，これらの最適な組み合わせ方が不明であるといった問題がある．

確率理論に基づいて統計情報を学習する研究，すなわち条件 **3** を満たす研究は数多く報告されている．ここで，入力文 A に対してその最も正しい構文構造 R を決定する方法として，式 (2.34) のように入力文 A と構文構造 R の同時確率 $P(R, A)$ を用いる研究 [28, 14, 17, 101, 37, 56, 57, 22, 23] と，式 (2.35) のように入力文 A が与えられたという条件の下で R を生成する条件付き確率 $P(R|A)$ を用いる研究 [16, 24, 25] の 2 つがある．

$$\arg\max_{R,A} \; P(R, A) \tag{2.34}$$

$$\arg\max_{R} \; P(R|A) \tag{2.35}$$

式 (2.36) に示すように，入力文 A が与えられているという条件の下では $P(A)$ は定数となるので，両者を用いた曖昧性解消は本質的には全く同一である．

$$\begin{aligned} \arg\max_{R} \; P(R, A) &= \arg\max_{R} \; P(A) \cdot P(R|A) \\ &= \arg\max_{R} \; P(R|A) \end{aligned} \tag{2.36}$$

両者の違いは，$P(R, A)$ の方が $P(R|A)$ よりも一般性が高いという点にある．すなわち，$P(R, A)$ から $P(R|A)$ を計算することはできるが，その逆はできない．これに加えて，入力 A が与えられているという条件が必要な条件付き確率 $P(R|A)$ に比べて，同時確率 $P(R, A)$ は入力 A が与えられない音声認識，文字認識などの曖昧性解消にも適用することができる．したがって，統計情報を確率モデルとして学習する際には，条件付き確率 $P(R|A)$ よりも同時確率 $P(R, A)$ を学習する方が望ましい．

本研究では，過去の研究における問題点をふまえ，以下のような方針で統計情報を利用した曖昧性解消を行う．

- 統合処理を行うことを前提に，複数の統計情報を利用して曖昧性解消を行う．
- 複数の統計情報を学習する際に，異なる種類の統計情報を個別に学習する．
- 個別に学習した統計情報を確率理論に基づいて組み合わせる．すなわち，個々の統計情報を反映したスコアがある事象の生成確率となり，またそれらの積が解析結果の候補全体の生成確率となるように個々の統計情報を学習する．
- 曖昧性解消に用いる確率モデルとして，入力文 A を条件としない同時確率を使用する．

また，これらの方針に基づき，解析結果の候補に対して種々の統計情報を反映したスコアを与える統合的確率言語モデルを提案する．次章では，この統合的確率言語モデルの詳細について述べる．

第3章　種々の統計情報を統合した日本語文解析

本章では，曖昧性解消に有効な種々の統計情報を同時に利用した日本語文解析について述べる．本研究における日本語文解析とは，形態素解析，構文解析，語義曖昧性解消を統合的に処理することを指す．もちろん，形態素解析，構文解析，語義曖昧性解消を行うことだけが自然言語解析の全てではないが，文脈・談話解析などの処理は本研究の対象外としている．

3.1 節では，形態素解析，構文解析，語義曖昧性解消の解析結果の候補に対してその生成確率を与える統合的確率言語モデルを提案する．この統合的確率言語モデルは，2.1 節で述べた 4 つの条件を満たしつつ，形態素解析，構文解析，語義曖昧性解消の問題解決に有効な統計情報を同時に利用できるように設計されている．そして，入力として与えられた日本語文の解析結果の候補に対して，統合的確率言語モデルによって与えられる生成確率の最も高い解析結果を選択することにより曖昧性を解消する．3.2 節では，3.1 節で提案した統合的確率言語モデルの評価実験について述べる．まず，コーパスを用いて統合的確率言語モデルの学習を行い，次に学習した統合的確率言語モデルを用いて日本語文の文節の係り受け解析実験を行う．さらに，解析に失敗した事例を調査することにより，統合的確率言語モデルの問題点を明らかにし，その対応策について述べる．

3.1　統合的確率言語モデル

まず，本論文で一貫して用いる記号について説明する．

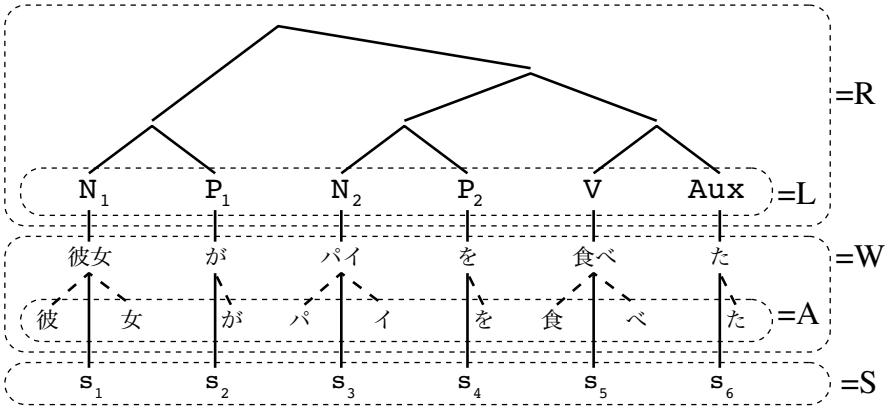
- 入力文字列 $A = a_1, \cdots, a_m$
- A を生成する単語列 $W = w_1, \cdots, w_n$
- W を生成する品詞列 $L = l_1, \cdots, l_n$
- L を生成する構文構造 R
- 単語 w_i の語義 s_i の集合 $S = s_1, \cdots, s_n$

例として,「彼女がパイを食べた」という入力文に対する解析結果の候補を図 3.1 に示す．

図 3.1 において，L 中の N, P, V, Aux はそれぞれ名詞，助詞，動詞，助動詞を表わす品詞，S 中の $s_1, \cdots, s_6$ は各単語 w_i の語義を表わしている．入力文字列 A に対してその W, L, R, S を求めることは，単語の区切りと品詞を決定する形態素解析，入力文の構文構造を決定する構文解析，各単語の語義を決定する語義曖昧性解消を同時に行うことを意味する．

一般に，1 つの入力文に対して図 3.1 のような解析結果の候補は複数存在する．そこで，各解析結果に対してその生成確率 $P(R, L, W, A, S)$ を計算し，その値の最も大きい解析結果を出力することによって曖昧性解消を行う．この生成確率 $P(R, L, W, A, S)$ を以下のように分解する．

$$P(R, L, W, A, S) = P(R) \cdot P(L|R) \cdot P(W|L, R) \cdot P(A|W, L, R) \cdot P(S|A, W, L, R) \tag{3.1}$$



$$
\begin{aligned}
A &= \text{彼}, \text{女}, \text{が}, \text{パ}, \text{イ}, \text{を}, \text{食}, \text{べ}, \text{た} \\
W &= \text{彼女}, \text{が}, \text{パイ}, \text{を}, \text{食べ}, \text{た} \\
L &= N_1, P_1, N_2, P_2, V, Aux \\
S &= s_1, \cdots, s_n
\end{aligned}
$$

図 3.1: 例文「彼女がパイを食べた」とその解析結果

ここで，構文構造 R は最終的に品詞列 L を生成するものと仮定すると，$P(L|R) = 1$ となる (図 3.1参照)．また，単語列 W が決まれば入力文字列 A は一意に決まるので，$P(A|W) = 1$ となる．したがって，式 (3.1) は以下のように簡略化できる．

$$P(R, L, W, A, S) = P(R) \cdot P(W|R) \cdot P(S|W, R) \tag{3.2}$$

本研究では，式 (3.2) に示した通り，解析結果の生成確率を以下の 3 つの確率モデルの積として計算する．

1. 構文モデル $P(R)$
   構文構造 R を生成する確率である．この構文モデルには，以下に示す統計情報をその確率値に反映させる．
   - 構文的優先度
   - 品詞連接に関する優先度
   - 距離に関する優先度
2. 語彙モデル $P(W|R)$
   構文構造 R が与えられたときに，それから単語列 W を生成する確率である．この語彙モデルには，以下に示す統計情報をその確率値に反映させる．
   - 単語の出現頻度
   - 語彙的従属関係
3. 語義モデル $P(S|R, W)$
   構文構造 R と単語列 W が与えられたときに，それから語義集合 S を生成する確率である．この語義モデルには，以下に示す統計情報をその確率値に反映させる．
   - 語義の出現頻度

- 語義の共起関係

以下，これら 3 つのサブモデルについて詳しく説明する．

3.1.1 構文モデル $P(R)$

構文モデルは構文構造 R の生成確率を与える確率モデルである．この構文モデルとしては，構文的な統計情報を反映し，かつ構文構造 R の生成確率を高い精度で推定するものであれば，どのような確率モデルを利用してもよい．このような確率モデルの例としては PCFG が挙げられる．すなわち，構文構造 R を作り出す構文規則の集合 $\{r_1, \cdots, r_n\}$ に対して，各規則 r_i の適用確率 $P(r_i)$ の積を計算することにより，構文モデル $P(R)$ の値を推定することができる．しかしながら，PCFG によって与えられる確率モデルには構文的優先度は反映されるものの，品詞連接に関する優先度は反映されない．また，2.6 節で述べたように，PCFG を拡張したモデルには距離に関する優先度が必ずしもうまく反映されないといった問題点もある．

確率一般化 LR 法 (Probabilistic Generalized LR Method，以下 PGLR と呼ぶ) [35, 36] もまた構文モデル $P(R)$ を与える確率モデルのひとつである．PGLR とは，CFG を利用して入力文の構文構造を決定する構文解析手法のひとつである GLR 法を拡張したものである．GLR 法は，まず CFG から LR 表と呼ばれる状態遷移オートマトンを作成し，この LR 表をもとに入力文に対して 1 つの状態遷移列を割り当てることにより，その構文構造を決定する．PGLR は，LR 表に記述された各状態遷移の遷移確率を推定し，その遷移確率の積によって 1 つの状態遷移列，すなわちそれに対応する構文構造 R の生成確率を与えるモデルである．GLR 法に確率を組み込む試みを最初に行ったのは Briscoe らである [12]．しかしながら，彼らの枠組によって計算される構文構造の生成確率は実際には構文構造の生成確率ではなく，これを構文モデル $P(R)$ としてそのまま用いることはできない．Inui らは，このような Briscoe らの手法の問題点を指摘し，R の生成確率 $P(R)$ を正しく与えるような新しい枠組を提案している [35, 36]．また，Sornlertlamvanich らは Briscoe らのモデルと Inui らのモデルを実験的に比較し，構文解析における曖昧性解消の精度は Inui らのモデルの方が優れていると報告している [98, 99, 100]．

本研究では，Inui らによる PGLR を構文モデル $P(R)$ の有力な候補として考える．なぜなら，PGLR は PCFG に比べて，以下のような特長を持っているからである [35, 36]．

- 文脈依存性を取り扱うことができる

  PCFG は文脈依存性を全く取り扱うことができないのに対し，PGLR は文脈依存性を若干取り扱うことができる．ここで，$l_1, \cdots, l_n$ を入力品詞列とし，その構文構造を GLR 法によって解析することを考える．このとき，l_{i-1} までの解析が終了し，そのときの状態が s_{i-1} である場合に，次の状態 s_i への遷移確率は以下のように計算される．

$$P(l_i, a_i | s_{i-1}) \tag{3.3}$$

$$P(a_i | s_{i-1}, l_i) \tag{3.4}$$

  l_i は先読み語と呼ばれ，l_{i-1} の直後に現われる品詞である．a_i はアクションと呼ばれるものであり，shift 動作と reduce 動作の 2 種類がある．式 (3.3) は現在の状態から先読み語 l_i と次のアクション a_i を予測する確率モデルであり，式 (3.4) は現在の状態と先読み語 l_i から次のアクション a_i を予測する確率モデルである．現在の状態 s_{i-1} が shift 動作の直後であれば式 (3.3) が，そうでなければ式 (3.4) が使われる [36]．いずれの場合も，s_{i-1}, l_i, a_i が特定できれば，次の状態 s_i は LR 表によって一意に決定されるので，これらの確率は状態 s_{i-1} から状態 s_i への遷移確率と考えてよい．このとき，状態 s_{i-1} は，既に解析されている品詞列 $l_1, \cdots, l_{i-1}$ を末端とする部分的な構文構造を抽象化したものであると考えられる．したがって，これらの遷移確率には左文脈がある程度反映されている．

- 品詞連接の優先度を取り扱うことができる

  式 (3.3) の確率モデルは，状態 s_{i-1} から次の先読み語 l_i を予測するモデルとなっている．また先ほど述べたように，状態 s_{i-1} は既に解析されている品詞列 $l_1, \cdots, l_{i-1}$ を末端とする部分的な構文構造を抽象化したものである．したがって，式 (3.3) の確率モデルには品詞 bi-gram $P(l_i|l_{i-1})$ に似た統計情報が反映されている．言い換えれば，式 (3.3) は品詞 bi-gram $P(l_i|l_{i-1})$ の確率の前件を LR 表の状態 s_{i-1} によって細分化したモデルである．したがって，構文モデル $P(R)$ の推定に PGLR を用いることによって，構文的優先度だけでなく，品詞 bi-gram のような品詞連接に関する優先度を同時に学習することができる．

- 距離に関する優先度を取り扱うことができる

  GLR 法における 2 つのアクションのうち，reduce 動作は現在までに解析されたある構成要素 A が他の構成要素 B に係ることを表わし，shift 動作は構成要素 A が構成要素 B には係らず，さらに遠くの要素に係ることを表わしている (図 3.2).

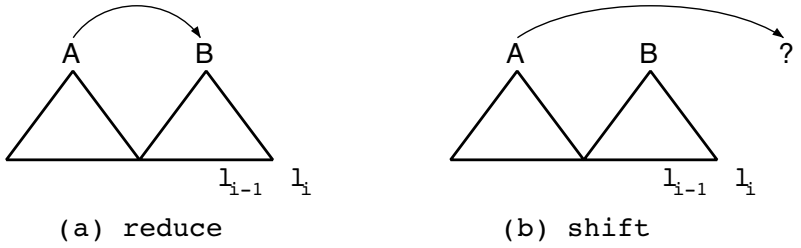


図 3.2: reduce 動作と shift 動作

  ここで，式 (3.3) 及び式 (3.4) は次の動作を予測する確率モデルとなっているので，日本語における「なるべく近い要素に係りやすい」といった距離に関する優先度を自然に組み込むことができる．すなわち，PGLR を実際に学習した場合には，reduce 動作を予測する確率の方が shift 動作を予測する確率よりも高くなると予想される．

以上のように，PGLR は構文モデル $P(R)$ として適した特性を持っている．構文モデルとして PGLR を用いる場合に問題となるのは，構文的優先度，品詞連接に関する優先度，距離に関する優先度が同時に学習されるということである．ところが，2.1 節で述べたように，それぞれの統計情報は独立に学習してから，これらを式 (3.5) のように組み合わせる方が望ましい．

$$P(R) \;=\; f(\; P_{syn}(R), P_{pos}(R), P_{dist}(R) \;) \tag{3.5}$$

式 (3.5) において，$P_{syn}(R)$, $P_{pos}(R)$, $P_{dist}(R)$ はそれぞれ構文的優先度，品詞連接に関する優先度，距離に関する優先度のみを反映させた統計量を表わす．しかしながら，現時点においては，これらの統計量を独立に学習し，かつそれらを組み合わせて構文構造 R の生成確率を与えるような枠組は発見されていない．また，3.1 節で述べたような構文構造 R が品詞列 L を生成するという仮定の下では，入力文における単語は全て品詞に抽象化されているため，PGLR の推定パラメタ空間はそれほど大きくなく，現時点で利用可能な言語資源からでも十分学習可能であると考えられる．

3.1.2 語彙モデル $P(W|R)$

語彙モデルは，品詞列 L を末端とする構文構造 R が与えられたときに，それから単語列 W を生成する確率である．但し，図 3.1 に示すように，W 中の各単語 w_i は，その品詞 l_i から生成されると仮定している．この語彙モデルは，式 (3.6) に示すように，各単語 w_i の生成確率の積として計算することができる．

$$P(W|R) = \prod_{w_i} P(w_i|R, w_1, \cdots, w_{i-1}) \tag{3.6}$$

例えば，図 3.1 の例において，単語を文の後ろから順番に生成していくと仮定すると，語彙モデル $P(W|R)$ は以下のような単語の生成確率の積として計算できる．

$$\begin{aligned}
P(W|R) &= P(\text{彼女}, \text{が}, \text{パイ}, \text{を}, \text{食べ}, \text{た}\ |R) && (3.7)\\
&= P(\text{た}\ |R) \cdot && (3.8)\\
& \quad P(\text{食べ}\ |R, \text{た}) \cdot && (3.9)\\
& \quad P(\text{を}\ |R, \text{食べ}, \text{た}) \cdot && (3.10)\\
& \quad P(\text{パイ}\ |R, \text{を}, \text{食べ}, \text{た}) \cdot && (3.11)\\
& \quad P(\text{が}\ |R, \text{パイ}, \text{を}, \text{食べ}, \text{た}) \cdot && (3.12)\\
& \quad P(\text{彼女}\ |R, \text{が}, \text{パイ}, \text{を}, \text{食べ}, \text{た}) && (3.13)
\end{aligned}$$

3.1.2.1 単語生成文脈

式 (3.6) の各項 (図 3.1 の例では式 (3.8)〜式 (3.13)) のパラメタ空間は非常に大きく，これを直接学習することは一般に不可能である．ところが，各単語 w_i の生成に強く影響するのは各項の確率の前件 $R, w_1, \cdots, w_{i-1}$ の全てではなく，その一部のみであると考えられる．例えば，図 3.1 の例文において,「パイ」は動詞「食べ」のヲ格の格要素となっている．このとき,「パイ」という単語を生成する際には，式 (3.11) の前件 $\langle R, \text{を}, \text{食べ}, \text{た} \rangle$ (図 3.3の斜線部) のうち，品詞 N と単語「を」,「食べ」(図 3.3の丸で囲まれた部分) が強く影響している．したがって，式 (3.11) を近似するひとつの方法として式 (3.14) が考えられる．

$$P(\text{パイ}\ |R, \text{を}, \text{食べ}, \text{た}) \simeq P(\text{パイ}\ |N[s(\text{食べ}, \text{を})]) \tag{3.14}$$

式 (3.14) において，$N[s(\text{食べ}, \text{を})]$ は,「食べ」という動詞のヲ格の格要素となっている名詞を表わしてい

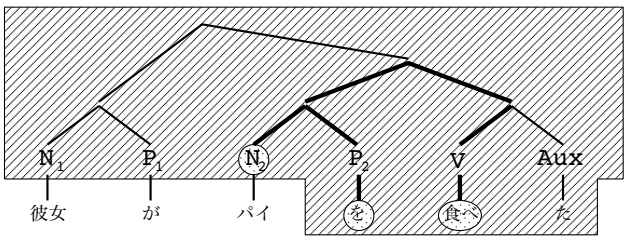


図 3.3: 「パイ」を生成するときの単語生成文脈

る．すなわち，$P(\text{パイ}\ |N[s(\text{食べ}, \text{を})])$ は,「食べ」という動詞のヲ格の格要素となっている名詞から「パイ」という単語が生成される確率を表わしている．したがって，式 (3.14) には,「パイ」という単語の出現頻度と,「パイ」と「食べ」の共起に関する語彙的従属関係が反映されている．

ここで，単語生成文脈 c_i を以下のように定義する．

> 単語 w_i の単語生成文脈 c_i とは，生成確率の前件 $R, w_1, \cdots, w_{i-1}$ から w_i の生成に強く影響する部分のみを取り出したものである．

先ほどの例においては，単語「パイ」の単語生成文脈は s(食べ, を) である．そして，各単語 w_i の生成確率のコンテキスト $R, w_1, \cdots, w_{i-1}$ を，その単語の品詞 l_i とその単語の単語生成文脈 c_i に抽象化し，語彙モデル $P(W|R)$ を以下のように近似推定する．

$$\begin{aligned} P(W|R) &= \prod_{w_i} P(w_i|R, w_1, \cdots, w_{i-1}) \\ &\simeq \prod_{w_i} P(w_i|l_i[c_i]) \end{aligned} \tag{3.15}$$

単語生成文脈という概念を導入したのは，各単語の生成確率のパラメタ空間を縮小することを目的としている．

先ほどの例では,「パイ」という単語を生成する際に，既に生成された 3 つの単語 { を, 食べ, た } の中から「食べ」と「を」を単語生成文脈の中に含めているが,「パイ」が助詞「を」を介して動詞「食べ」に係るということは構文構造 R によって決められている．このように，単語生成文脈 c_i は，w_i の生成確率の前件 $R, w_1, \cdots, w_{i-1}$ 全体を抽象化したものである．

3.1.2.2 単語生成文脈決定規則

3.1.2.1 で述べたように，単語生成文脈とは，各単語の生成確率の前件からその単語の生成に強く影響する部分を切り出して抽象化したものである．ここで，どのような単語に対してどのような単語生成文脈を選ぶかが問題となる．単語生成文脈を決定する際には，以下の点に注意しなければならない．

1. 単語の生成確率 $P(w_i|l_i[c_i])$ が既存の言語資源から十分学習可能であること
2. 単語の生成確率に曖昧性解消に有効な統計情報が反映されていること

単語生成文脈を決定する際に，特に 2. の条件を満たすためには，人間が持つ言語学的な知識を利用することが望ましい．3.1.2.1 で挙げた例において,「パイ」の単語生成文脈を s(食べ, を) に決定する際には，格要素となる名詞が助詞を介して動詞に係る場合には,「食べるという動詞のヲ格には食べ物を表わす単語が現われやすい」というように，それらの単語の間に強い従属関係があるという言語学的知識を利用している．そこで本研究では，言語学的知見に基づくヒューリスティクス規則によって単語生成文脈を決定する．以下，単語 w_i の単語生成文脈 c_i を決定する規則を**単語生成文脈決定規則**と呼ぶ．

単語生成文脈として何を選択するかを自動的に学習することも考えられる．例えば，2.5 節で述べたように，Magerman は確率の前件としてどのような素性を選択すればよいのかを決定木を用いて自動学習している [57]．しかしながら，本研究では人手で記述された規則によって単語生成文脈を選択するアプローチを取る．なぜなら，単語生成文脈を用いて単語の生成確率を近似する際に，近似された単語の生成確率が現在利用可能な言語資源から十分学習可能であるか否かを考慮することができるからである．すなわち，単語生成文脈を人手で決定することによって，曖昧性解消に用いるモデルの複雑さをコントロールすることができる．また，語彙モデルにどのような種類の語彙的従属関係を反映させるかを単語生成文脈決定規則によって明確に記述することにより，モデルに反映された統計情報が曖昧性解消に有効であるかどうかなど，モデルの特性の分析を容易に行えることも，単語生成文脈を人手によって決定する理由のひとつである．

以下，単語生成文脈決定規則の例をいくつか挙げる．

- 語彙的従属関係を全く考慮しない場合

  単語 w_i について，周囲の単語との従属関係を考慮しない場合には，その単語の生成確率はその単語の品詞 l_i にのみ依存するとみなす．例えば，図 3.1 の例において，助動詞「た」と動詞「食べ」を生成する際に他の単語との語彙的従属関係を考えない場合には，それぞれの生成確率 (3.8),(3.9) は以下のように近似できる．

$$P(\text{た}\,|R) \simeq P(\text{た}\,|AUX) \tag{3.16}$$

$$P(\text{食べ}\,|R, \text{た}) \simeq P(\text{食べ}\,|V) \tag{3.17}$$

  このとき，これらの生成確率には語彙的従属関係は反映されず，単語の出現頻度のみが反映される．

  助動詞「た」，動詞「食べ」の生成確率の前件を図示したのが図 3.4，図 3.5 である．これらの図において，斜線部が近似する前の生成確率の前件，丸で囲まれた部分が近似された生成確率の前件を表わしている．

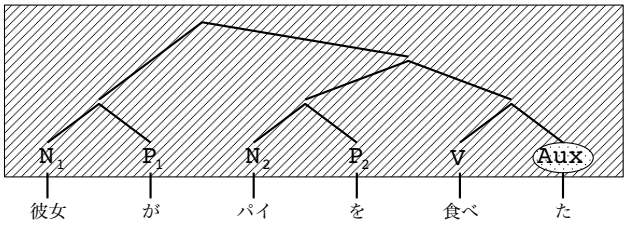


図 3.4: 「た」を生成するときの単語生成文脈

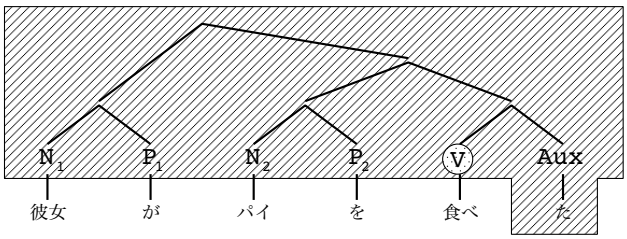


図 3.5: 「食べ」を生成するときの単語生成文脈

  周囲の単語との語彙的従属関係を考慮しない場合の単語生成文脈は，単語生成文脈決定規則 #1 によって決定される．この規則は単語生成文脈を決定する際のデフォルト規則でもある．

  【単語生成文脈決定規則 #1】

  > 単語 w_i を生成する際に他の単語との従属関係を考慮しない場合には，単語 w_i の単語生成文脈 c_i を空とする．このとき，w_i の生成確率 $P(w_i|l_i)$ は，品詞 l_i から単語 w_i が生成される確率を表わす．

- 格要素となる名詞が助詞を介して動詞に係っている場合

  格要素となる名詞が助詞を介して動詞に係る際には，動詞・助詞の組と名詞との間には語彙的従属関係が存在する．この語彙的従属関係を確率モデルに反映させるために，以下のような単語生成文脈決定規則を定義する．

【単語生成文脈決定規則 #2】

単語 w_i の品詞 l_i が N(名詞) であり，かつ助詞 p を介して動詞 v に係っているとき，単語 w_i の単語生成文脈 c_i を $s(v,p)$ とする．このとき，w_i の生成確率 $P(w_i|N[s(v,p)])$ は動詞 v の格 p の格要素となる名詞 N から単語 w_i が生成される確率を表わす．

例えば，図 3.1 の例において，名詞「パイ」は動詞「食べ」のヲ格の格要素であり，名詞「彼女」は動詞「食べ」のガ格の格要素となっている．したがって，これらの単語を生成する際にはこの規則が適用され，それぞれの生成確率 (3.11),(3.13) は以下のように近似される．

$$P(\text{パイ}\,|R,\text{を},\text{食べ},\text{た}) \simeq P(\text{パイ}\,|N[s(\text{食べ},\text{を})]) \tag{3.18}$$

$$P(\text{彼女}\,|R,\text{が},\text{パイ},\text{を},\text{食べ},\text{た}) \simeq P(\text{彼女}\,|N[s(\text{食べ},\text{が})]) \tag{3.19}$$

また，名詞「パイ」,「彼女」の生成確率の前件を図示したのが図 3.3，図 3.6 である．斜線部が近似する前の生成確率の前件，丸で囲まれた部分が近似された生成確率の前件を表わしている．

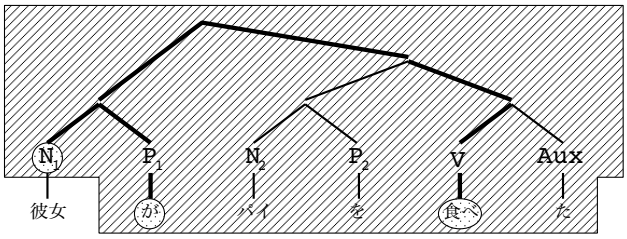


図 3.6: 「彼女」を生成するときの単語生成文脈

- 助詞とその係り先用言の従属関係，格間の従属関係を考慮する場合

助詞とその係り先となる用言との間には語彙的従属関係が存在する．例えば，助詞「へ」は動詞「行く」に係りやすいが,「泳ぐ」という動詞には係りにくい．また，助詞「を」は他動詞に係ることはあるが，自動詞や形容詞に係ることはない．

これに加えて，同じ用言に係る助詞の間にも語彙的従属関係が存在する．例えば，助詞「まで」が動詞「飛ぶ」に係っているとき,「から」という助詞も同じ動詞「飛ぶ」に係ることが多い．(ex.「旅客機が東京からニューヨークまで飛ぶ」)．また，同じ助詞が同じ動詞に係る，いわゆる二重格のような現象は起こりにくい．

以上から，用言に係る助詞を生成する際には，助詞とその係り先用言との間の語彙的従属関係や，同じ用言に係る助詞同士の従属関係 (以下，これを格間の従属関係と呼ぶ) を考慮することが望ましい．

図 3.1 の例文では,「が」と「を」の 2 つの助詞が動詞「食べ」に係っている．これらの助詞の生成確率 (3.10),(3.12) を以下のように近似する．

$$P(\text{を}\,|R,\text{食べ},\text{た}) \simeq P(\text{を}\,|P[m(\text{食べ},2,\{\phi_1,\phi_2\})]) \tag{3.20}$$

$$P(\text{が}\,|R,\text{パイ},\text{を},\text{食べ},\text{た}) \simeq P(\text{が}\,|P[m(\text{食べ},2,\{\phi_1,\text{を}\})]) \tag{3.21}$$

式 (3.20) は，助詞 P が 2 つの助詞の係り先となっている動詞「食べ」に係っているときに，品詞 P から単語「を」が生成される確率を表わしている．これにより，助詞「を」が動詞「食べ」にどの程

度係りやすいかといった，助詞とその係り先用言との間の語彙的従属関係が確率モデルに反映される．一方，式 (3.21) は，助詞 P が 2 つの助詞の係り先となりかつそのうちの 1 つは「を」である動詞「食べ」に係っているときに，品詞 P から単語「が」が生成される確率を表わしている．これにより，助詞「が」とその係り先の動詞「食べ」の間の語彙的従属関係だけでなく，助詞「が」と助詞「を」が動詞「食べ」にどの程度同時に係りやすいかといった格間の従属関係も確率モデルに反映される．

助詞「を」,「が」の生成確率の前件を図示したのが図 3.7，図 3.8 である．斜線部が近似する前の生成確率の前件，丸で囲まれた部分が近似された生成確率の前件を表わす．

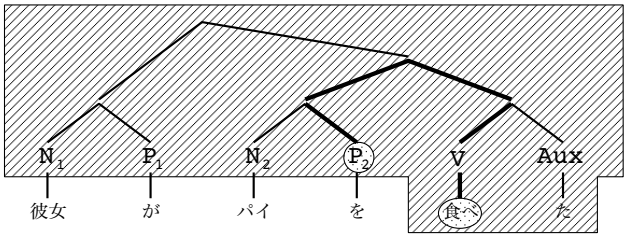


図 3.7: 「を」を生成するときの単語生成文脈

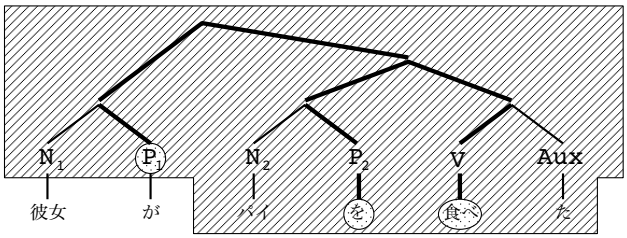


図 3.8: 「が」を生成するときの単語生成文脈

助詞とその係り先用言の従属関係および格間の従属関係を助詞の生成確率に反映し，かつその推定パラメタ空間を縮小するために，単語生成文脈決定規則 #3 を以下のように定義する．

【単語生成文脈決定規則 #3】

> 単語 w_i の品詞 l_i が P(助詞) でありかつ用言 h に係っているとき，単語 w_i の単語生成文脈 c_i を $m(h, n, \{\phi_1, \cdots, \phi_j, p_{j+1}, \cdots, p_n\})$ とする．このとき，w_i の生成確率 $P(w_i|P[m(h, n, \{\phi_1, \cdots, \phi_j, p_{j+1}, \cdots, p_n\})])$ は，用言 h が n 個の助詞の係り先となりかつ用言に近い $p_{j+1}, \cdots, p_n$ の助詞が既に生成されているときに，ϕ_j として w_i が生成される確率を表わす．

単語生成文脈決定規則 #3 において，同じ用言に係る助詞は用言に近いものから順番に生成されると仮定している．すなわち，助詞が出現する順序も考慮されている．また，用言 h に係る助詞の数 n が常に確率モデルの前件に加えられているのは，n が構文構造 R から一意に決まるためである．

同じ用言 h に係る助詞の生成確率の積は，複数の助詞を同時に生成する確率モデルとしても計算で

きる．

$$\prod_{j=n}^{1} P(p_j|P[m(h,n,\{\phi_1,\cdots,\phi_j,p_{j+1},\cdots,p_n\})]) \tag{3.22}$$

$$= \quad P(p_1,\cdots,p_n|P_1\cdots P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})]) \tag{3.23}$$

式 (3.23) は，$P_1,\cdots,P_n$ の n 個の助詞が同じ用言 h に係るときに，それらから n 個の単語 $p_1,\cdots,p_n$ を同時に生成する確率を表わしている．例えば，図 3.1 の例において，同じ動詞「食べ」に係る助詞「が」と「を」の生成確率の積，すなわち式 (3.20) と式 (3.21) の積は，2 つの助詞 P_1 と P_2 が動詞「食べ」に係るときに単語「が」と「を」がこの順序で同時に生成される確率 (式 (3.25)) に等しい．

$$P(\text{が}\,|P_1[m(\text{食べ},2,\{\phi_1,\text{を}\})])\cdot P(\text{を}\,|P_2[m(\text{食べ},2,\{\phi_1,\phi_2\})]) \tag{3.24}$$

$$= \quad P(\text{が},\text{を}\,|P_1,P_2[m(\text{食べ},2,\{\phi_1,\phi_2\})]) \tag{3.25}$$

- 助詞の係り先が用言か否かを考慮する場合

  助詞の係り先が用言である場合とそうでない場合には，その生成確率 $P(w_i|P)$ の分布は著しく異なる．例えば，助詞の係り先が用言の場合には「が」,「を」などの助詞は出現しやすいが，助詞「の」は出現しにくい．これに対して，係り先が体言の場合，すなわちその助詞を含む文節が連体修飾節となっている場合には，助詞「の」が出現する場合が圧倒的に多い．したがって，助詞の生成確率 $P(w_i|P)$ を学習する際に，その助詞の係り先が用言もしくは体言であるかを区別しないで学習するのは望ましくない．

  例えば，以下の例文において，文節「彼-の」は文節「写真-を」に係っている．

  彼-の / 写真-を / 見-た[1]

  このとき,「の」の生成確率 $P(\text{の}\,|P)$ は，助詞 P が体言に係っているので，かなり高い値を持つと考えられる．しかしながら，用言に係る助詞と体言に係る助詞の両方から確率分布 $P(w_i|P)$ を学習した場合には，$P(\text{の}\,|P)$ の値が低く見積られる．

  そこで，以下のような単語生成文脈決定規則を定義する．

  【単語生成文脈決定規則 #4】

  > 単語 w_i の品詞 l_i が P(助詞) であり，かつその助詞の係り先が体言であるとき，単語 w_i の単語生成文脈 c_i を nd とする．nd はその助詞の係り先が体言であることを表わすシンボルである．このとき，w_i の生成確率 $P(w_i|P[nd])$ は，体言を係り先とする助詞から単語 w_i が生成される確率を表わす．

  単語生成文脈決定規則 #3 が用言に係る助詞の生成確率を推定する際に適用されるのに対し，単語生成文脈決定規則 #4 は体言に係る助詞の生成確率を推定する際に適用される．

- 副詞的名詞の係り先に関する従属関係を考慮する場合

  名詞は動詞の格要素となるだけではなく，用言を修飾し副詞的に働くことがある．このような名詞は副詞的名詞と呼ばれる．例えば，以下の 4 つの例文を見てみよう．

[1] “-” は単語の区切りを，“/” は文節の区切りを表わす．

(a)　三月-、 / 東京-を / 訪れ-た
(b)　三月 / 東京-を / 訪れ-た
(c)　三月-、 / 三日-は / ひなまつり-だ
(d)　三月 / 三日-は / ひなまつり-だ

上記の例文において，名詞を主辞とする「三月-、」および「三月」のような文節は用言にも体言にも係ることができる．例えば，例文 (a),(b) においては,「三月」はともに動詞「訪れ」を連用修飾しているのに対し，例文 (c),(d) においては,「三月」はともに名詞「三日」を連体修飾している．言いかえれば，例文 (a),(b) においては「三月」は副詞的名詞として働き，例文 (c),(d) においては「三月」は複合名詞の構成要素となっている．このような名詞を主辞とする文節は，用言にも体言にも係る可能性があるので係り先文節の候補数も多くなり，正しい係り先を特定することが難しい．

ここで,「三月」のような日時を表わす名詞が連用修飾する場合には，(a) のように読点「、」を伴うことが多く，(b) のように読点「、」を伴わない場合は少ない．これに対し，日時を表わす名詞が連体修飾する場合には，(c) のように読点「、」を伴うことは少なく，(d) のように読点「、」を伴わない場合が多い．このように，その名詞が連用修飾するのか連体修飾するのかといった修飾関係の種類や，そのときに読点を伴うのか伴わないのかといった違いは，曖昧性解消に有効な情報となりうる．

以上を考慮し，名詞の生成確率を計算する際の単語生成文脈決定規則を以下のように定義する．

【単語生成文脈決定規則 #5】

> 単語 w_i の品詞 l_i が N(名詞) であり，かつその名詞が用言の格要素ではないとき，単語 w_i の単語生成文脈 c_i を $(touten, mod_type)$ とする．$touten$ は名詞 w_i が同じ文節内に読点を伴っているか否かを表し，読点を伴っているときには “、”，そうでないときには “ϕ” といった値を取る．一方，mod_type は名詞 w_i とその係り先要素との修飾関係を表わし，名詞 w_i が連用修飾するなら “連用”，連体修飾するなら “連体” といった値を取る．このとき，w_i の生成確率 $P(w_i|N[(touten, mod_type)])$ は，条件 $(touten, mod_type)$ の下で単語 w_i が生成される確率を表わしている．

例えば，例文 (a) においては,「三月-、」という文節が「東京-を」に係るという解釈と「訪れ-た」に係るという解釈がある．このとき，それぞれの解釈に対して，名詞「三月」の生成確率は以下のように近似される．

– 「三月-、」が「東京-を」に係るとき　$P(\text{三月}\,|N[(\text{、}, \text{連体})])$
– 「三月-、」が「訪れ-た」に係るとき　$P(\text{三月}\,|N[(\text{、}, \text{連用})])$

このとき，$P(\text{三月}\,|N[(\text{、}, \text{連体})]) < P(\text{三月}\,|N[(\text{、}, \text{連用})])$ であると考えられるので,「三月-、」が「訪れ-た」に係るという正しい解釈に高い確率が与えられる．例文 (d) についても同様のことが言える．

– 「三月」が「三日-は」に係るとき　$P(\text{三月}\,|N[(\phi, \text{連体})])$
– 「三月」が「ひなまつり-だ」に係るとき　$P(\text{三月}\,|N[(\phi, \text{連用})])$

このとき，$P(\text{三月}\,|N[(\phi, \text{連体})]) > P(\text{三月}\,|N[(\phi, \text{連用})])$ であると考えられるので,「三月」が「三日-は」に係るという正しい解釈に高い確率が与えられる．

以上に挙げた単語生成文脈決定規則 #1〜#5 が単語生成文脈を決定するための全ての規則というわけではない．ここでは，特に用言の格関係に注目して語彙モデルに反映させるべき語彙的従属関係の例を挙げ

たが，他の種類の語彙的従属関係を語彙モデルに反映させるように単語生成文脈決定規則を拡張・洗練することもできる．このように，語彙モデルにおいてどのような語彙的従属関係を考慮するかは，単語生成文脈決定規則の追加・変更によって柔軟に調整することができる．

ここで注意しなければならないのは，単語生成文脈決定規則によって決定される単語生成文脈の中に生成する単語 w_i 以外の単語が含まれている場合，その単語は w_i の前に生成されていなければならないということである．例えば，単語生成文脈決定規則 #2 によって決定される単語生成文脈の中には，動詞 v と助詞 p が含まれる．したがって，動詞 v と助詞 p は名詞 w_i の前に生成されていなければならない．一般に，入力文中の全ての単語 w_i について，単語生成文脈を用いて w_i の生成確率を近似するためには，以下の制約が満たされていなければならない．

【単語の生成順序に対する制約】

語彙モデル $P(W|R)$ において単語列 $W = w_1, \ldots, w_n$ を生成する際には，単語 w_i の単語生成文脈に含まれる単語は w_i よりも必ず前に生成されていなければならない．

語彙モデルを推定する際に，単語生成文脈決定規則 #1〜#5 のみを用いる場合には，各規則によって定められる単語生成文脈に含まれる単語は生成する単語 w_i よりも必ず右に位置しているため，単語を文の後ろから順番に生成していけば上記の制約は満たされる．また，単語生成文脈決定規則を追加・変更する際には，上記の制約を満たす単語の生成順序が存在することを考慮に入れなければならない．

3.1.2.3 従属係数

これまでは単語を生成する際に考える単語生成文脈は常に一つであると仮定していた．しかしながら，一般には，1 つの単語を生成する際に複数の単語生成文脈を考慮しなければならない場合もある．例えば，並列構造を持つ文中において，ある要素が 2 つの要素に同時に係る場合などが考えられる．このような並列構造を持った例文を図 3.9 に挙げる．

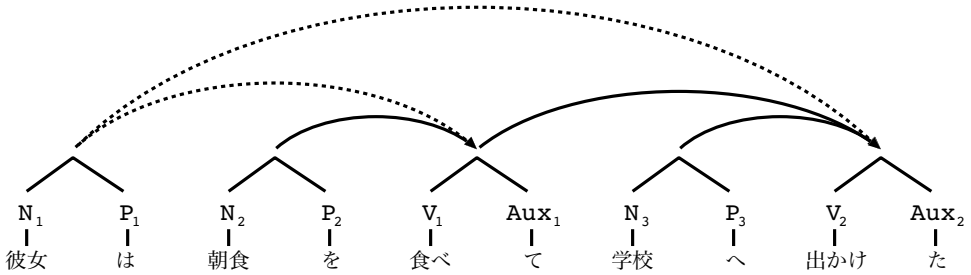


図 3.9: 並列構造を持つ例文

図 3.9 の例文において，2 つの文節「食べ-て」と「出かけ-た」は並列の関係にある．したがって，この例文中の名詞「彼女」は動詞「食べ」のハ格の格要素であり[2]，同時に動詞「出かけ」のハ格の格要素でもある．したがって，単語生成文脈決定規則 #2 に従えば,「彼女」という単語を生成する際の単語生成文脈として s(食べ, は) と s(出かけ, は) の 2 つがあると考えられる．このとき,「彼女」の生成確率を次のように近似するのが妥当である．

$$P(\text{彼女}\,|N[s(\text{食べ}, \text{は}), s(\text{出かけ}, \text{は})]) \tag{3.26}$$

[2] 本研究では，名詞が助詞を介して用言に係る場合は常に，その名詞を用言の表層格の格要素とみなしている．

式 (3.26) の確率は，ある名詞が動詞「食べ」のハ格の格要素であり，かつ動詞「出かけ」のハ格の格要素でもあるという条件の下で「彼女」が生成される確率を表わしている．同様に，この例文中の助詞「は」は動詞「食べ」と「出かけ」の両方に係っている．したがって，単語生成文脈決定規則 #3 に従えば,「は」という単語を生成する際の単語生成文脈として $m(\text{食べ}, 2, \{\phi_1, \text{を}\})$ と $m(\text{出かけ}, 2, \{\phi_1, \text{へ}\})$ の 2 つがあると考えられ,「は」の生成確率も次のように推定される．

$$P(\text{は}\,|P[m(\text{食べ}, 2, \{\phi_1, \text{を}\}), m(\text{出かけ}, 2, \{\phi_1, \text{へ}\})]) \tag{3.27}$$

式 (3.27) の確率は，ある助詞が動詞「食べ」と「出かけ」の両方に係り，しかも動詞「食べ」に係る 2 つの助詞のうちひとつが「を」でありかつ動詞「出かけ」に係る 2 つの助詞のうちひとつが「へ」であるという条件の下で,「は」が生成される確率を表わしている．

このように，1 つの単語を生成する際に複数の単語生成文脈を考えなければならない場合もある．ところが，式 (3.26) や式 (3.27) のように複数の単語生成文脈を同時に前件に持つ確率モデルは，推定するパラメタの数が爆発的に増大する．

そこで本研究では，複数の単語生成文脈を以下のように取り扱う．まず，説明を簡略化するために，単語 w_i が 2 つの単語生成文脈 c_1 と c_2 を持つとする．このとき，単語 w_i の生成確率 $P(w_i|l_i[c_1, c_2])$ を以下のように変形する．

$$\begin{aligned}
P(w_i|l_i[c_1, c_2]) &= \frac{P(l_i[c_1, c_2]|w_i) \cdot P(w_i)}{P(l_i[c_1, c_2])} && (3.28)\\
&= \frac{P(l_i[c_1]|w_i) \cdot P(l_i[c_2]|l_i[c_1], w_i) \cdot P(w_i)}{P(l_i[c_1]) \cdot P(l_i[c_2]|l_i[c_1])} && (3.29)\\
&\simeq \frac{P(l_i[c_1]|w_i) \cdot P(l_i[c_2]|l_i, w_i) \cdot P(w_i)}{P(l_i[c_1]) \cdot P(l_i[c_2]|l_i)} && (3.30)\\
&= \frac{P(l_i[c_1]|w_i)}{P(l_i[c_1])} \cdot \frac{P(l_i[c_2]|l_i, w_i)}{P(l_i[c_2]|l_i)} \cdot P(w_i) && (3.31)\\
&= \frac{P(w_i|l_i[c_1])}{P(w_i)} \cdot \frac{P(w_i, |l_i, l_i[c_2])}{P(w_i|l_i)} \cdot P(w_i) && (3.32)\\
&= P(w_i|l_i) \cdot \frac{P(w_i|l_i[c_1])}{P(w_i|l_i)} \cdot \frac{P(w_i|l_i[c_2])}{P(w_i|l_i)} && (3.33)
\end{aligned}$$

式 (3.29) から式 (3.30) の変形において，2 つの単語生成文脈 c_1 と c_2 は互いに独立であると仮定している．

$$\begin{aligned}
P(l_i[c_2]|l_i[c_1]) &\simeq P(l_i[c_2]|l_i) && (3.34)\\
P(l_i[c_2]|l_i[c_1], w_i) &\simeq P(l_i[c_2]|l_i, w_i) && (3.35)
\end{aligned}$$

先ほどの例で言えば，ある名詞が単語生成文脈 $s(\text{食べ}, \text{は})$ と取ることと $s(\text{出かけ}, \text{は})$ を取ることは互いに独立であると仮定している．

ここで，従属係数 $D(w_i|l_i[c_i])$ を式 (3.36) のように定義する．

$$D(w_i|l_i[c_i]) = \frac{P(w_i|l_i[c_i])}{P(w_i|l_i)} \tag{3.36}$$

この従属係数を用いれば，式 (3.33) から式 (3.38) が導かれる．

$$\begin{aligned}
P(w_i|l_i[c_1, c_2]) &\simeq P(w_i|l_i) \cdot \frac{P(w_i|l_i[c_1])}{P(w_i|l_i)} \cdot \frac{P(w_i|l_i[c_2])}{P(w_i|l_i)} && (3.37)\\
&= P(w_i|l_i) \cdot D(w_i|l_i[c_1]) \cdot D(w_i|l_i[c_2]) && (3.38)
\end{aligned}$$

以上の式変形は単語 w_i が 2 つの単語生成文脈を持つ場合を考えていたが，単語 w_i が n 個の単語生成文脈 $c_1, \cdots, c_n$ を持つ場合にも同様の式変形が可能であり，最終的に以下の式が得られる．

$$P(w_i|l_i[c_1, \cdots, c_n]) \simeq P(w_i|l_i) \cdot \prod_{c_i} D(w_i|l_i[c_i]) \tag{3.39}$$

式 (3.36) で定義した従属係数 $D(w_i|l_i[c_i])$ は単語 w_i と単語生成文脈 c_i の相関関係の強さを表わす統計量である．その性質を以下に述べる．

- w_i と c_i に相関関係がない場合，すなわち w_i と c_i が互いに独立である場合には，式 (3.36) の分子 $P(w_i|l_i[c_i])$ は分母 $P(w_i|l_i)$ にほぼ等しくなる．その結果，従属係数は 1 に近い値を取る．

$$D(w_i|l_i[c_i]) \simeq 1$$

- w_i と c_i に正の相関関係がある場合，すなわち w_i と c_i が共起しやすい場合には，単語生成文脈 c_i を前件に加えた確率 $P(w_i|l_i[c_i])$ は単語生成文脈 c_i を無視した確率 $P(w_i|l_i)$ よりも大きくなるので，従属係数は 1 より大きい値を取る．

$$D(w_i|l_i[c_i]) \gg 1$$

- w_i と c_i に負の相関関係がある場合，すなわち w_i と c_i が共起しにくい場合には，単語生成文脈 c_i を前件に加えた確率 $P(w_i|l_i[c_i])$ は単語生成文脈 c_i を無視した確率 $P(w_i|l_i)$ よりも小さくなるので，従属係数は 1 より小さく，0 に近い値を取る．

$$D(w_i|l_i[c_i]) \ll 1$$

したがって，複数の単語生成文脈 $c_1, \cdots, c_n$ の下での単語 w_i の生成確率は，単語生成文脈を無視した単語の生成確率 $P(w_i|l_i)$ と，w_i と c_i の相関関係を他の単語生成文脈とは独立に評価した従属係数 $D(w_i|l_i[c_i])$ の積によって計算できることを式 (3.39) は示している．従属係数 $D(w_i|l_i[c_i])$ を他の単語生成文脈と独立に評価することにより，確率モデルのパラメタ空間を推定可能な大きさに抑制することができる．

図 3.9 の例において,「彼女」の生成確率 (3.26) と「は」の生成確率 (3.27) はそれぞれ以下のように計算される．

$$\begin{aligned} & P(\text{彼女}\,|N[s(\text{食べ}, \text{は}), s(\text{出かけ}, \text{は})]) \\ \simeq\ & P(\text{彼女}\,|N) \cdot D(\text{彼女}\,|N[s(\text{食べ}, \text{は})) \cdot D(\text{彼女}\,|N[s(\text{出かけ}, \text{は})]) \end{aligned} \tag{3.40}$$

$$\begin{aligned} & P(\text{は}\,|P[m(\text{食べ}, 2, \{\phi_1, \text{を}\}), m(\text{出かけ}, 2, \{\phi_1, \text{へ}\})]) \\ \simeq\ & P(\text{は}\,|P) \cdot D(\text{は}\,|P[m(\text{食べ}, 2, \{\phi_1, \text{を}\})]) \cdot D(\text{は}\,|P[m(\text{出かけ}, 2, \{\phi_1, \text{へ}\})]) \end{aligned} \tag{3.41}$$

式 (3.40) において，$D(\text{彼女}\,|N[s(\text{食べ}, \text{は})])$ は「彼女」という単語が「食べ」という動詞のハ格の格要素としてどの程度出現しやすいのかを示しており，$D(\text{彼女}\,|N[s(\text{出かけ}, \text{は})])$ は「彼女」という単語が「出かけ」という動詞のハ格の格要素としてどの程度出現しやすいのかを示している．また，式 (3.41) において，$D(\text{は}\,|P[m(\text{食べ}, 2, \{\phi_1, \text{を}\})])$ は「は」という単語が「食べ」という動詞にどの程度係りやすいのか，またそのとき同じ動詞「食べ」に係る助詞として「を」とどの程度共起しやすいのかを示している．$D(\text{は}\,|P[m(\text{出かけ}, 2, \{\phi_1, \text{へ}\})])$ も同様の意味を持つ．

従属係数という概念を導入することにより，1 つの単語の生成確率を計算する際に複数の単語生成文脈を取り扱うことができることは既に述べた．従属係数を導入するもうひとつの利点として，式 (3.44) に示す

ように，語彙モデル $P(W|R)$ を単語の出現頻度のみを反映したモデル $P_{cf}(W|L)$ と語彙的従属関係のみを反映したモデル $D(W|R)$ との積に分解できるという点が挙げられる．

$$P(W|R) \simeq \prod_i P(w_i|l_i[C_{w_i}]) \tag{3.42}$$

$$\simeq \prod_{w_i} P(w_i|l_i) \cdot \prod_{c_{ij} \in C_{w_i}} D(w_i|l_i[c_{ij}]) \tag{3.43}$$

$$= P_{cf}(W|L) \cdot D(W|R) \tag{3.44}$$

$$P_{cf}(W|L) = \prod_{w_i} P(w_i|l_i) \tag{3.45}$$

$$D(W|R) = \prod_{w_i} \prod_{c_{ij} \in C_{w_i}} D(w_i|l_i[c_{ij}]) \tag{3.46}$$

上式において，C_{w_i} は単語 w_i の単語生成文脈の集合を表わしている．

式 (3.45) のモデル $P_{cf}(W|L)$ は単語生成文脈を無視したときに品詞 l_i から単語 w_i が生成される確率の積を表わしており，単語の出現頻度が反映される．これに対し，式 (3.46) のモデル $D(W|R)$ は各単語 w_i とその単語生成文脈 c_{ij} の従属係数の積を表わしており，w_i と c_{ij} の相関関係に関する優先度 (語彙的従属関係) が反映される．このように，語彙モデルを 1 つの統計情報のみを反映した 2 つのモデルの積として分解することができるため，2.1 節で述べたように，曖昧性解消時におけるそれぞれの統計情報の働きを容易に理解することができる．

3.1.3 語義モデル $P(S|W,R)$

語義モデルは，構文構造 R 及び単語列 W が与えられたときに，それから各単語 w_i の語義 s_i の集合 S が生成される確率である．この語義モデルは，語彙モデルと同様に，以下のように推定できる．

$$P(S|R,W) \simeq P_{cfs}(S|W) \cdot D_s(S|R,W) \tag{3.47}$$

$$P_{cfs}(S|W) = \prod_{s_i} P(s_i|w_i) \tag{3.48}$$

$$D_s(S|R,W) = \prod_{s_i} \prod_{sc_{ij} \in C_{s_i}} D(s_i|w_i[sc_{ij}]) \tag{3.49}$$

$$D_s(s_i|w_i[sc_{ij}]) = \frac{P(s_i|w_i[sc_{ij}])}{P(s_i|w_i)} \tag{3.50}$$

式 (3.48) のモデル $P_{cfs}(S|W)$ は単語 w_i から語義 s_i が生成される確率の積を表わしている．したがって，このモデルには語義の出現頻度が反映される．これに対し，式 (3.49) のモデル $D_s(S|R,W)$ は各語義 s_i とその語義生成文脈 sc_{ij} の従属係数の積を表わしている．語義生成文脈 sc_{ij} とは，単語生成文脈と同様に，語義 s_i の生成確率の前件からその語義の生成に最も影響のある部分のみを取り出し，抽象化したものである．語義の従属係数 $D_s(s_i|w_i[sc_{ij}])$ は，語彙モデルのときと同様に式 (3.50) で定義される．この従属係数には，s_i と sc_{ij} の相関関係 (語義の共起に関する優先度) が反映される．また，単語生成文脈決定規則と同様に，どのような語義 s_i に対してどのような語義生成文脈 sc_{ij} を選択するかをルールベースで記述することにより，確率モデルに組み込む語義の共起に関する統計情報の種類を容易に選択することができる．

以上のように，本研究で提案する統合的確率言語モデルは，語義の曖昧性解消にも対応できる枠組となっている．しかしながら，実際に語義モデルを学習するのは現時点では難しい．その最も大きな理由として，語義モデルを学習するのに十分な量の言語資源が存在しないことが挙げられる．語義モデルを学習するには少なくとも各単語の語義が付与されたコーパスが必要である．現在，EDR コーパス [67] など，そのような語義が付加されたコーパスも存在するが，語義モデルの推定には語彙モデル以上に推定するパラメタ空間が膨大であり，確率モデルとして学習できるほど十分なデータ量がない．本研究においては，語義曖昧性解消を形態素解析や統語解析と同時に行う枠組のみを提案し，これを実現することは今後の課題とする．

3.1.4 本節のまとめ

本節で提案した統合的確率言語モデルを概要を以下にまとめる．

- 形態素解析，構文解析，語義曖昧性解消を統合的に処理し，その解析結果のスコアとして R，L，W，A，S の同時確率 $P(R, L, W, A, S)$ を用いることにより曖昧性解消を行う．
- 式 (3.2) に示したように，同時確率 $P(R, L, W, A, S)$ を構文モデル $P(R)$，語彙モデル $P(W|R)$，語義モデル $P(S|R, W)$ の 3 つの確率モデルに分解する．
- 構文モデル $P(R)$ は構文構造 R の生成確率である．この構文モデルとして PGLR モデルを使用した場合，以下に挙げる統計情報を同時に学習することができる．
  - 構文的優先度
  - 品詞連接に関する優先度
  - 距離に関する優先度
- 語彙モデル $P(W|R)$ は単語列 W の生成確率である．この語彙モデルを，式 (3.44) に示したように，2 つのサブモデル $P_{cf}(W|L)$ と $D(W|R)$ の積に分解する．また，各サブモデルにおいて以下の統計情報を学習する．
  - 単語の出現頻度
    単語生成文脈を無視した単語の生成確率の積 $P_{cf}(W|L)$ に反映される．
  - 語彙的従属関係
    単語と単語生成文脈との間の相関関係を表わした従属係数の積 $D(W|R)$ に反映される．
- 語義モデル $P(S|R, W)$ は語義集合 S の生成確率である．この語義モデルを，式 (3.47) に示したように，2 つのサブモデル $P_{cfs}(S|W)$ と $D_s(S|R, W)$ の積に分解する．また，各サブモデルにおいて以下の統計情報を学習する．
  - 語義の出現頻度
    語義生成文脈を無視した語義の生成確率の積 $P_{cfs}(S|W)$ に反映される．
  - 語義の共起関係に関する優先度
    語義と語義生成文脈との間の相関関係を表わした従属係数の積 $D_s(S|R, W)$ に反映される．

図 3.10 は，図 3.1 に示した例文とその解析結果の候補に対して，統合的確率言語モデルによって与えられる解析結果の生成確率がどのような要素から計算されるかを例示したものである．但し，簡単のため語義モデル $P(S|R, W)$ については省略している．

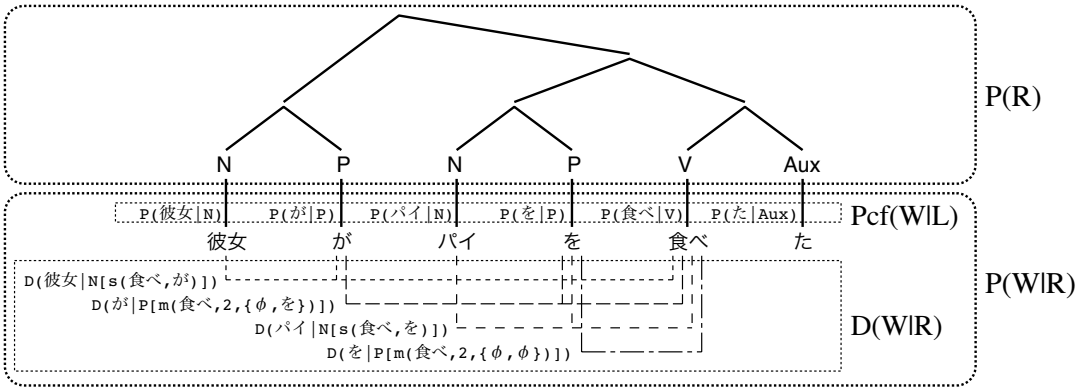


図 3.10: 統合的確率言語モデル

統合的確率言語モデルでは，生成確率 $P(R, W)$ が以下の確率もしくは従属係数の積によって計算されることを図 3.10 は示している．

- $P(R)$
  構文的な統計情報を反映した構文モデル
- $P(\text{彼女}\,|N)$, $P(\text{が}\,|P)$, $P(\text{パイ}\,|N)$, $P(\text{を}\,|P)$, $P(\text{食べ}\,|V)$, $P(\text{た}\,|Aux)$
  単語の出現頻度を反映した各単語の生成確率
- $D(\text{彼女}\,|N[s(\text{食べ}, \text{が})])$, $D(\text{パイ}\,|N[s(\text{食べ}, \text{を})])$
  格要素となる名詞と動詞・助詞の組との間の語彙的従属関係を反映した従属係数
- $D(\text{が}\,|N[m(\text{食べ}, 2, \{\ \phi\ , \text{を}\ \})])$, $D(\text{を}\,|N[m(\text{食べ}, 2, \{\ \phi\ , \phi\ \})])$
  助詞とその係り先となる動詞との間の従属関係，及び格間の従属関係を反映した従属係数

異なる種類の統計情報は異なる確率もしくは従属係数に反映されているため，個々の統計情報を個別に学習することができるだけでなく，個々の統計情報が曖昧性解消にどのように働くかを容易に分析することができる．このように，異なる種類の統計情報を独立に取り扱う点が，本研究で提案する統合的確率言語モデルの大きな特徴である．

3.2 評価実験

本節では，3.1 節で提案した統合的確率言語モデルの評価実験について述べる．本節の実験では，統合的確率言語モデルの中で取り扱う様々な統計情報の中でも特に語彙的従属関係に注目する．2.5 節で述べた通り，近年の統計的自然言語処理に関する研究においても，解析結果の候補に与えるスコアに語彙的従属関係を反映させて解析精度を向上させる研究が主流になっている．本研究では，語彙的従属関係を確率モデルに反映させるという点ではこれらの多くの先行研究と同じであるが，語彙的従属関係を従属係数の積 $D(W|R)$ (式 (3.46)) に局所化したことにその特長がある．しかしながら，語彙的従属関係を他の統計情報と独立に取り扱うこのようなアプローチが妥当かどうかは実験的に確かめる必要がある．そのためには，3.1 節で提案した統合的確率言語モデルは形態素解析，構文解析，語義曖昧性解消を同時に行うことを前提としている

が，これらを同時に行うよりも構文解析，特に語彙的従属関係が曖昧性解消に最も有効に働くと考えられる係り受け解析のみを行った方が，語彙的従属関係の取り扱い方が適切かどうかを調べるという目的に適っている．また，3.1.3 項で述べたように，語義モデル $P(S|R,W)$ を実際に学習するのは現時点では難しく，語義曖昧性解消を他の解析と同時に行うことはできない．このような理由から，本節では予備実験として，文節列を入力とした文節間の係り受け解析を行う．

まず，3.2.1 項で構文モデルの学習について述べ，次に 3.2.2 項で語彙モデルの学習について述べる．学習した構文モデル，語彙モデルを用いた日本語文の文節の係り受け解析実験については 3.2.3 項で述べる．最後に 3.2.4 項で実験結果の考察を行う．

3.2.1 構文モデルの学習

本節の実験で行う日本語文解析は文節の係り受け解析である．すなわち，入力として単語列，品詞列，文節区切りが与えられたときに，それぞれの文節の係り先文節を決定する．このような文節の係り受け解析を CFG を用いて行った．例として,「先月，彼女の兄は結婚した」という例文について，その例文の構文構造を図 3.11 に，またこれを作り出す CFG 規則を図 3.12 に示す．

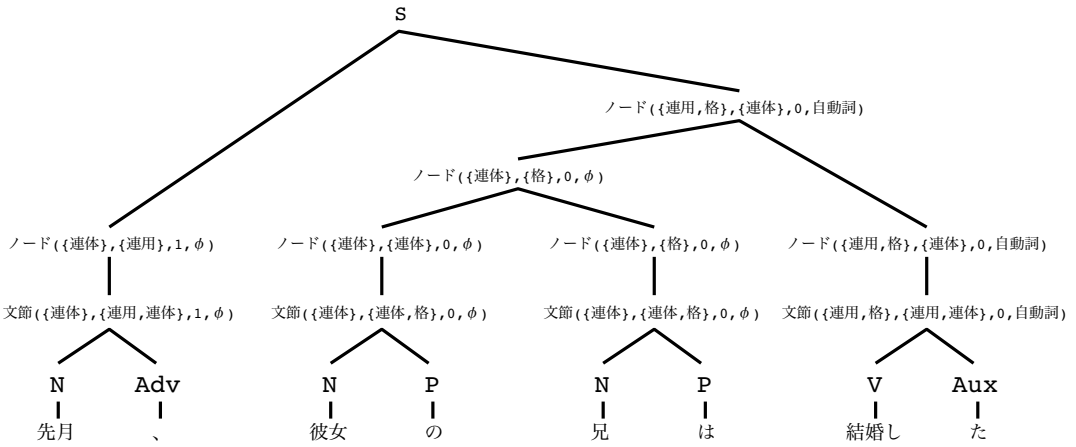


図 3.11: CFG 規則によって与えられる構文構造の例

S	→ ノード ({連体},{連用},1,ϕ)	ノード ({連用, 格},{連体},0, 自動詞)	(1)
ノード ({連用, 格},{連体},0, 自動詞)	→ ノード ({連体},{格},0,ϕ)	ノード ({連用, 格},{連体},0, 自動詞)	(2)
ノード ({連体},{格},0,ϕ)	→ ノード ({連体},{連体},0,ϕ)	ノード ({連体},{格},0,ϕ)	(3)
ノード ({連体},{連用},1,ϕ)	→ 文節 ({連体},{連用, 連体},1,ϕ)		(4)
ノード ({連体},{連体},0,ϕ)	→ 文節 ({連体},{連体, 格},0,ϕ)		(5)
ノード ({連体},{格},0,ϕ)	→ 文節 ({連体},{連体, 格},0,ϕ)		(6)
ノード ({連用},{連体},0, 自動詞)	→ 文節 ({連用},{連用, 連体},0, 自動詞)		(7)

図 3.12: 図 3.11 の構文構造を作り出す CFG 規則

3.2.1.1 文節ラベル

図 3.12 に示した CFG 規則の終端記号 (“文節” で始まるシンボル) は，文節の統語的特性を反映したシンボルである．以下，これを文節ラベルと呼ぶ．文節ラベルの定義を (3.51) に示す．

$$\text{文節ラベル} \stackrel{def}{=} \text{文節}\ (\{\text{受け属性}\}, \{\text{係り属性}\}, \text{読点の有無}, \text{用言種別}) \tag{3.51}$$

ここで，“受け属性”，“係り属性” は，それぞれ文節の受け属性と係り属性を表わし，その文節タイプによって表 3.1 のように定められる．

表 3.1: 文節タイプとその受け属性，係り属性

文節タイプ	受け属性	係り属性
動詞*	連用，格	連用，連体
形容詞*	連用，格	連用，連体
名詞述語	連用，連体，格	連用，連体
動詞+助詞	連用，格	連体，格
形容詞+助詞	連用，格	連体，格
名詞述語+助詞	連用，格	連体，格
指示詞*	連用	連用，連体
副詞*	連用	連用
感動詞*	連用	連用
連体詞*	連用	連体
名詞*	連体	連用，連体
サ変名詞*	連体	連用，連体
後置詞文節	連体	連体，格
接続詞*	ϕ	連用
サ変名詞+読点	連用，連体，格	連用，連体

ここで文節タイプとは，文節を構成する単語の品詞情報をもとに決定される文節の細分類である．この文節タイプについては後ほど詳説する．

表 3.1 において，受け属性はその文節がどのような修飾を受けるかを表わしている．すなわち，文節が連用修飾を受ける場合にはその受け属性は “連用” であり，連体修飾を受ける場合には “連体” である．また，文節が連用修飾を受ける場合でも，その文節が用言を含み，その用言の格要素を表わす文節の修飾を受ける場合には，その受け属性を “連用” ではなく “格” として区別する[3]．最後に，その文節が何も修飾を受けない場合にはその受け属性を “ϕ” とする．例えば，表 3.1 において，文節タイプが “名詞” の文節は他の文節の連体修飾のみを受けることを表わし，文節タイプが “動詞” の文節は他の文節の連用修飾を受けたりその格要素を表わす他の文節の修飾を受けることを表わす．一方，係り属性は，その文節がどのような修飾を行うかを表わしている．“連用” はその文節が他の文節を連用修飾することを，“連体” は連体修飾することを表わし，“格” はその文節が格要素として用言に係ることを表わしている．例えば，表 3.1 において，文節タイプが “名詞” の文節は他の文節を連用修飾もしくは連体修飾することを表わし，文節タイプが “副詞” の文節は他の文節を必ず連用修飾することを表わす．

[3] ここでの “格” とは用言を受け側とした格関係のみを指す．

次に，文節の細分類である文節タイプの詳細を以下に示す．

動詞など

表 3.1 において，末尾に “*” のついた文節タイプは，その文節において最も主要な意味を持つ単語の品詞によって決められる．具体的には，品詞を名詞・動詞・形容詞などの自立語と助詞・助動詞・接尾辞などの付属語とに分類し，文節中で最も右に位置する自立語を文節タイプとする．以下に例を挙げる．

例 1	動詞 (買っ)	助動詞 (た)		→	文節タイプ=動詞
例 2	名詞 (先月)	接尾辞 (末)	記号 (、)	→	文節タイプ=名詞

後置詞文節

主に名詞と助詞から構成されている文節である．その例を以下に挙げる．

例 1	名詞 (警視庁)	名詞 (幹部)	助詞 (の)
例 2	名詞 (太郎)	助詞 (は)	読点 (、)

後置詞文節は,「警視庁-幹部-の」が「失態-が」に係るときのように連体修飾することもあり，また「太郎-は-、」が「出かけ-た」に係るときのように用言を修飾してその用言の格要素を表わすこともあるので，係り属性は “連体，格” となっている．

名詞述語

名詞述語を主辞とする文節であり，主に形容動詞 (例 1) もしくは名詞と判定詞 (例 2) から構成されている．

例 1	形容動詞 (画期的だ)	
例 2	名詞 (先生)	判定詞 (だ)

この文節は，動詞や形容詞を主辞とする文節と同様に “連用” や “格” を受け属性として持つ他に,「私-の」が「先生-だ」に係るときのように連体修飾も受けることができるため，“連体” も受け属性として持つ．

動詞+助詞，形容詞+助詞，名詞述語+助詞

主に動詞，形容詞，名詞述語と助詞から構成される文節である．

例 1	動詞 (走る)	名詞 (の)	助詞 (は)
例 2	形容詞 (安い)	助詞 (と)	助詞 (は)
例 3	形容動詞 (画期的な)	名詞 (の)	助詞 (は)

これらの文節は，文節タイプが “動詞”，“形容詞”，“名詞述語” である文節と同様に “連用” や “格” を受け属性として持つ．また，後置詞文節と同様に，“連体” や “格” を係り属性として持つ．

サ変名詞+読点

主にサ変名詞と読点から構成されている文節である．

例　　サ変名詞　読点
　　(賛成)　(、)

(a)　意見-は / 賛成-、 / 反対-に / 分かれ-た

(b)　最初-は / 賛成-、 / その-後 / 反対-した

これらの文節は，(a) のようにサ変名詞が名詞として働くときには受け属性として “連体” を持ち，(b) のようにサ変名詞が動詞として働くときには受け属性として “連用” と “格” を持つ．

以上で述べた文節タイプは，文節を構成する単語の品詞情報によって一意に決定されるものとする．また，文節タイプが決まれば，文節の受け属性と係り属性は表 3.1 によって一意に決定される．

次に，文節タイプの定義式 (3.51) における受け属性と係り属性以外の要素について説明する．“読点の有無” はその文節の末尾が読点であれば “1”，そうでなければ “0” といった値を取る．日本語においては，文節はなるべく近い文節に係る傾向があるが，読点を末尾に持つ文節は直後の文節には係りにくく，読点を末尾に持たない文節よりも遠くに係る傾向を持っている．素性 “読点の有無” を文節ラベルに付加したのは，読点を末尾に持つ文節と持たない文節に異なる文節ラベルを与えることにより，両者の距離に関する優先度の違いを確率モデルに反映させるためである．一方 “用言種別” は，“格” を受け属性に含む文節タイプを細分化するための属性であり，文節に含まれる用言が自動詞，他動詞，形容詞，名詞述語のときにはそれぞれ “自動詞”，“他動詞”，“形容詞”，“名詞述語” といった値を取る．また，“格” を受け属性に持たない文節に対してはその値は常に “ϕ” である．3.1.2 項で定義した単語生成文脈決定規則は，語彙的従属関係の中でも特に用言の格関係に注目している．“格” を受け属性に持つ文節の文節ラベルを細分化したのはこのためである．

3.2.1.2　文節の係り受け解析を行う CFG

本実験で用いた CFG 規則は，文節の係り属性を一意に決める規則と，文節間の係り受けに関する制約を表わす規則の 2 種類に分けられる．ここでは，これらの CFG 規則の詳細，および CFG の作成手順について説明する．

図 3.12 中の規則 (4)〜(7) は文節の係り属性を一意に決める規則である．例えば，図 3.11 中の後置詞文節「彼女-の」は「兄-は」を連体修飾しているので，規則 (5) が適用される．なぜなら，この規則の左辺の非終端記号に含まれる係り属性は { 連体 } のみであり，複数の係り属性の候補 (この場合 “連体” と “格”) を持つ文節の係り属性が実際には “連体” であることを表わしているからである．一方，図 3.11 中の後置詞文節「兄-は」は動詞を主辞とする文節「結婚-した」に係り，動詞 “結婚” の格要素を表わしているので，規則 (6) が適用される．なぜなら，この規則の左辺の非終端記号に含まれる係り属性は { 格 } のみであり，文節の実際の係り属性が “格” であることを表わしているからである．このように，後置詞文節がどれだけ連体修飾しやすいのか，もしくは他の文節の格要素となるのかといった統計情報は，規則 (5),(6) が使われたときの確率の差として反映される．すなわち，文節の係り属性に曖昧性がある場合に，どの係り属性が選ばれやすいのかといった統計情報が確率モデルに反映される．

一方，図 3.12 中の規則 (1)〜(3) は，文節間の係り受けに関する制約を表わしている．例えば，規則 (2) は，係り属性が “格” である文節 (もしくはそれを主辞とする句) が，受け属性が “連用，格” である文節 (もしくはそれを主辞とする句) に係ることができることを表わしている．逆に，係り属性が “格” である文節 (もしくはそれを主辞とする句) が，受け属性が “連体” である文節 (もしくはそれを主辞とする句) に係ることはないので，以下のような規則は存在しない．

ノード ({連体},{連体},0, 自動詞)　→　ノード ({連体},{格},0,ϕ)　ノード ({連体},{連体},0, 自動詞)

このように，木構造を作り出す CFG の各規則は，受け属性と係り属性の整合性をチェックする働きを持つ．また，これらの規則において，左辺の非終端記号と右辺の一番右にある非終端記号は常に等しい．なぜなら，日本語における句の主辞は常に一番右に位置する文節であり，句全体の統語的特性 (受け属性，係り属性など) はその主辞の統語的特性が継承されるからである．

本研究では，文節の係り受け解析を行うこのような CFG をコーパスから自動的に作成した．CFG の作成には京大コーパス [47] を使用した．この京大コーパスの各例文には，単語区切り，単語の品詞，文節区切りと文節の係り受け解析の結果が付加されている．この京大コーパスの 9,944 例文に対して以下の操作を行うことにより，文節の係り受け解析を行う CFG を作成した．

1. 文節の係り受け解析の結果を図 3.11 のような木構造に置き換える．
2. 文節の品詞並びから文節ラベルを決定する．
3. 文節ラベルの一段上にある非終端記号のラベルを決定する．文節の係り属性の候補が複数ある場合にはその文節の正しい係り属性を決めなければならないが，図 3.11 に示すような木構造により文節の係り先が一意に決まるため，文節の正しい係り属性も一意に決定できる．
4. 残りの非終端記号のラベルを決定する．すなわち，一番右端にある子ノードのラベルを親ノードに継承させる．
5. ルートノードに開始記号 “S” を与える．
6. 木構造の枝分れを CFG 規則に変換する．

このようにして得られた文法の概要を表 3.2 に示す．

表 3.2: 文法の概要

規則数	961
非終端記号数	51
終端記号数 (文節ラベル数)	42

3.2.1.3 構文モデルのパラメタ推定

本実験においては，構文モデル $P(R)$ として PGLR を利用する．3.1.1 項で述べたように，PGLR は，ある構文構造を生成する状態遷移列に対してそれぞれの状態遷移確率の積を計算し，その値を構文構造の生成確率 $P(R)$ とする．したがって，LR 表における状態遷移確率を推定パラメタとし，これらを訓練データから学習しなければならない．まず，訓練コーパスの各例文とそれに付加された構文構造を作り出す状態遷移列を求め，また状態遷移が行われた回数を数え上げる．このようにして得られた状態遷移回数を式 (3.3),(3.4) のような遷移確率に変換することにより，確率モデル $P(R)$ のパラメタ推定を行った．本実験においては，以上の手順により，CFG を作成したのと同じ京大コーパス 9,944 文を用いて構文モデルの学習を行った．

3.2.2 語彙モデルの学習

本項では語彙モデルの学習について述べる．まず，語彙モデルの式 (3.44) を以下に再掲する．

$$P(W|R) \quad = \quad P_{cf}(W|L) \cdot D(W|R) \tag{3.52}$$

$$P_{cf}(W|L) = \prod_i P(w_i|l_i) \tag{3.53}$$

$$D(W|R) = \prod_{w_i} \prod_{c_{ij} \in C_{w_i}} D(w_i|l_i[c_{ij}]) \tag{3.54}$$

本実験においては，$P_{cf}(W|L)$ の計算を省略できる．なぜなら，単語区切り及び各単語の品詞はすでに入力として与えられているため，全ての解析結果の候補について品詞から単語への生成確率の積 $P_{cf}(W|L)$ は等しいからである．したがって，語彙モデルとして学習するのは従属係数の積 $D(W|R)$ のみでよい．

従属係数 $D(w_i|l_i[c_{ij}])$ は，生成される単語 w_i および単語生成文脈 c_{ij} の種類によって学習する統計情報が異なるために，学習方法および学習に用いる言語資源の種類も異なる．以下，それぞれの従属係数の学習について順番に説明する．

3.2.2.1 格要素に関する従属係数の学習

単語生成文脈決定規則 #2 によって，ある名詞 n の単語生成文脈が $s(v,p)$ であると定められたときには，その従属係数 (3.55) が語彙モデルに反映される．

$$D(n|N[s(v,p)]) = \frac{P(n|N[s(v,p)])}{P(n|N)} \tag{3.55}$$

この従属係数を学習するために，RWC コーパス [75] と EDR 共起辞書 [67] から，名詞 n が助詞 p を介して動詞 v に係る事例 (n,p,v) を収集した．RWC コーパスは毎日新聞 5 年分 (1991〜1995 年) の新聞記事を収集した品詞タグ付きコーパスである．したがって，単語の品詞はコーパスに付加されているものの，単語間の係り受け関係は付加されていないので，コーパス中の名詞がどの動詞に係っているのかについては正確にはわからない．しかしながら,「文の最後に “名詞 助詞 動詞” といった品詞列がある場合にはそれらは係り受け関係にあるとみなすことができる」などのいくつかのヒューリスティクスを用いて事例 (n,p,v) の収集を行った [102]．その結果，のべ 6,888,849 組の事例を収集した．一方，EDR 共起辞書は EDR コーパスから様々なタイプの共起データを抽出した事例集であり，その中には (n,p,v) といったタイプの事例も含まれている．そこで，EDR 共起辞書に記載されたのべ 975,510 個の事例も訓練事例に加えた．これらの訓練事例を用いて，式 (3.55) の分子および分母の確率モデルは以下のように最尤推定できる．

$$P(n|N[s(v,p)]) \simeq \frac{O(n,p,v)}{\sum_n O(n,p,v)} \tag{3.56}$$

$$P(n|N) \simeq \frac{O(n)}{\sum_n O(n)} \tag{3.57}$$

但し，$O(n,p,v)$ は事例 (n,p,v) の出現頻度を表わし，$O(n)$ は名詞 n の出現頻度を表わす．

RWC コーパスと EDR 共起辞書からのべ 800 万近くの事例を収集したが，確率モデル $P(n|N[s(v,p)])$ を推定するために十分な学習データ量が得られたとはいえない．そこで，確率モデル $P(n|N[s(v,p)])$ を推定する際に，以下のような近似を行った．

- 名詞 n の意味クラスによる抽象化

  格要素となる名詞 n を名詞クラスを用いて抽象化することにより推定パラメタ空間を縮小する．すなわち，名詞 n の意味クラスの集合を $C_n = \{c_{n_1}, \cdots, c_{n_m}\}$ として，従属係数 $D(n|N[s(v,p)])$ を以下のように計算した．

$$D(n|N[s(v,p)]) \simeq \frac{\sum_j P(n|c_{n_j}) P(c_{n_j}|N[s(v,p)])}{P(n|N)} \tag{3.58}$$

$P(c_{n_j}|N[s(v,p)])$ は，動詞 v の格 p の格要素として名詞意味クラス c_{n_j} が生成される確率を表わし，$P(n|c_{n_j})$ は，名詞意味クラス c_{n_j} から名詞 n が生成される確率を表わしている．これらの確率モデルを以下のように最尤推定した (但し，簡単のため c_{n_j} を c_n に置きかえている).

$$P(c_n|N[s(v,p)]) \simeq \frac{O(c_n,p,v)}{\sum_{c_n} O(c_n,p,v)} \tag{3.59}$$

$$P(n|c_n) \simeq \frac{O(n,c_n)}{\sum_n O(n,c_n)} \tag{3.60}$$

ここで，$O(c_n,p,v)$ はある名詞意味クラス c_n が動詞 v の格 p の格要素となっている事例の数であり，また $O(n,c_n)$ は名詞 n の意味クラスが c_n であるという事例の数である．

$O(c_n,p,v)$ は，事例集合 (n,p,v) の名詞 n を意味クラスに抽象化することによって数えあげた．但し，名詞 n が複数の意味クラス $\{c_{n_1},\cdots,c_{n_m}\}$ に属する場合は，事例 (c_{n_j},p,v) $(1 \leq j \leq m)$ の頻度を $1/m$ として数えあげた．また，このとき名詞 n を名詞意味クラス c_n に抽象化した回数を同様に数えあげ，それを $O(n,c_n)$ とした．

本実験においては，名詞意味クラス c_n として日本語語彙体系 [30, 31] の名詞シソーラスを利用した．この名詞シソーラスは 2,666 個の意味クラスから成り，それらの上位下位関係は木構造によって表わされている．また，シソーラスに登録されている名詞数は 264,861 である．この日本語語彙体系は固有名詞が数多く登録されていることに特長がある．本研究では，名詞シソーラスのルートから深さ 3 に位置する 151 個の意味クラスの集合を名詞クラスとして用いた．これらの意味クラスは互いに排他的である．

$D(n|N[s(v,p)])$ を式 (3.58) によって推定する場合，名詞 n が日本語語彙体系に登録されておらず，その名詞意味クラス c_n が不明な場合には，その従属係数は学習不可能であるとして，$D(n|N[s(v,p)]) \simeq 1$ とした．これは，n と $s(v,p)$ との間の語彙的従属関係を無視することに相当する．

- バックオフ方式によるスムージング

  確率モデル $P(c_n|N[s(v,p)])$ は，c_n が動詞 v の格 p の格要素となっている事例の集合 (c_n,p,v) により最尤推定する．しかしながら，この訓練事例の数が十分にない場合には，それによって推定された確率モデルの信頼性も低いと考えられる．そこで，確率モデルの分母となる事例 $(*,p,v)$ ($*$ は任意の名詞意味クラスを表わす) の数がある閾値 λ よりも小さい場合には，v を動詞意味クラス c_v を用いて抽象化した確率モデル $P(c_n|N[s(c_v,p)])$ によって近似した．

$$P(c_n|N[s(v,p)]) \simeq P(c_n|N[s(c_v,p)]) \tag{3.61}$$

  また，$(*,p,c_v)$ の事例数が λ を越えない場合には，動詞意味クラス c_v の抽象度を段階的に上げていき，必ず λ 個以上の事例から確率モデルを推定するようにした．

  本実験では，動詞意味クラス c_v として分類語彙表 [109] を利用した．分類語彙表の動詞シソーラスには 16,524 個の動詞が登録されている．この分類語彙表の 5 桁および 2 桁の分類コードを動詞意味クラスとして利用した[4]．動詞を分類語彙表の 2 桁の分類コードに抽象化しても学習事例数が λ を越えなかったとき，もしくは $(*,p,v)$ の事例数が λ 以下でありかつ動詞 v が分類語彙表に登録されていなかった場合には，十分信頼度の高い確率モデルが学習できなかったとして，$D(w_i|N[s(v,p)]) \simeq 1$ とした．これは，n と $s(v,p)$ との間の語彙的従属関係を無視することに相当する．なお，本実験では $\lambda = 100$ とした．

[4] 分類語彙表においては，桁数の少ない分類コードほど抽象度の高い意味クラスを表わす．

また，動詞 v が受身形または使役形である場合には，格要素に関する従属係数を 1 とした．

$$v \text{ が受身形または使役形のとき} \qquad D(w_i|N[s(v,p)]) \simeq 1 \tag{3.62}$$

これは，動詞が受身形・使役形である場合とそうでない場合とでは，名詞が同じ助詞を介して動詞に係っていても，名詞と動詞の統語的関係が異なる場合があるためである．例えば,「彼が書いた」という文においては，助詞「が」は名詞「彼」が動詞「書く」の主格の格要素であることを表わすが，動詞が受身形となっている「手紙が書かれた」という文においては，助詞「が」は名詞「手紙」が動詞「書く」の目的格の格要素であることを表わす．動詞が使役形である場合にも同様のことが言える．ここで用いた訓練共起データ (n,p,v) は，動詞 v が受身形でも使役形でもない事例を収集したものであるので，このデータを基に推定した従属係数 $D(w_i|N[s(v,p)])$ を動詞が受身形もしくは使役形である場合に用いるのは適切ではない．動詞が受身形もしくは使役形であるときの訓練共起データ (n,p,v) を収集し，それから従属係数を推定するべきである．しかしながら，今回の実験では，動詞が受身形もしくは使役形である場合の従属係数を推定するのに十分な訓練共起データを収集することができなかったため，$D(w_i|N[s(v,p)])$ を 1 とし，格要素に関する語彙的従属関係を無視した．

3.2.2.2 用言に係る助詞に関する従属係数の学習

単語生成文脈決定規則 #3 によって，ある助詞 p_i の単語生成文脈が $m(h,n,\{\phi_1,\cdots,\phi_i,p_{i+1},\cdots,p_n\})$ であると定められたときには，その従属係数 (3.63) が語彙モデルに反映される．

$$D(p_i|P[m(h,n,\{\phi_1,\cdots,\phi_i,p_{i+1},\cdots,p_n\})]) = \frac{P(p_i|P[m(h,n,\{\phi_1,\cdots,\phi_i,p_{i+1},\cdots,p_n\})])}{P(p_i|P)} \tag{3.63}$$

n 個の助詞 $p_1,\cdots,p_n$ が用言 h に同時に係る場合には，それぞれの p_i に対応する従属係数の積を計算すれば良い．これらの従属係数の積は式 (3.66) のように変形できる[5]．

$$\prod_i D(p_i|P[m(h,n,\{\phi_1,\cdots,\phi_{i-1},p_i,\cdots,p_n\})]) \tag{3.64}$$

$$= \prod_i \frac{P(p_i|P[m(h,n,\{\phi_1,\cdots,\phi_i,p_{i+1},\cdots,p_n\})])}{P(p_i|P)} \tag{3.65}$$

$$= \frac{P(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})])}{\prod_i P(p_i|P)} \tag{3.66}$$

$$\stackrel{def}{=} D(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})]) \tag{3.67}$$

したがって，学習しなければならないのは，n 個の助詞 $P_1,\cdots,P_n$ がある用言 h に係っているときに，単語 $p_1,\cdots,p_n$ を同時に生成する確率モデル $P(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})])$ と，品詞 P(助詞) から単語 p_i を生成する確率モデル $P(p_i|P)$ である．以降，簡単のため，前者の確率モデルを以下のように記述する．

$$P(\vec{p} \mid h,n) \stackrel{def}{=} P(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})]) \tag{3.68}$$
$$\text{但し，} \vec{p} = (p_1,\cdots,p_n)$$

ここでは式 (3.66) の分子の確率モデル $P(\vec{p} \mid h,n)$ の学習についてのみ説明し，式 (3.66) の分母の各項 $P(p_i|P)$ の推定については 3.2.2.3 で説明する．

[5] 式 (3.66) の分子の導出については式 (3.23) を参照.

確率モデル $P(\vec{p}|h,n)$ を学習するために，n 個の助詞 $\vec{p}$ が用言 h に同時に係るという事例 $(\vec{p},h)$ を EDR コーパス [67] の全ての例文 (201,339 例文) から収集した．EDR コーパスの各例文には括弧付けによる構文構造が付加され，この構文構造により主辞 h に係る助詞の組 $\vec{p}$ は特定されるので，事例の収集は自動的に行うことができる．今回の実験では，用言 h として動詞，形容詞，名詞述語の 3 つを考えた．主辞 h が動詞，形容詞，名詞述語であるときの，また h に係る助詞の数 n が 1，2，3，4 以上であるときの収集した事例 $(\vec{p},h)$ ののべ数を表 3.3 にまとめる．

表 3.3: EDR コーパスから収集した事例 $(\vec{p},h)$ ののべ数

h	$n=1$	$n=2$	$n=3$	$n\geq 4$
動詞	231,730	123,915	30,375	3,961
形容詞	19,266	7,686	1,292	154
名詞述語	28,636	9,327	1,238	98

n が 4 以上のときには学習に十分な事例を収集することができなかった．そこで，n が 4 以上のときには，従属係数を 1，すなわち助詞とその係り先用言との語彙的従属関係や格間の従属関係を無視した．

$$n \geq 4 \text{ のとき} \qquad D(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})]) \ \simeq \ 1 \tag{3.69}$$

また，先ほど述べたように，用言 h が動詞でありかつ受身形または使役形である場合には，それ以外の場合と区別して従属係数を学習すべきだが，動詞が受身形・使役形であるときの訓練共起データ $(\vec{p},h)$ を十分収集することができなかったため，同様に従属係数を 1 とした．

用言 h が動詞で受身形または使役形のとき

$$D(p_1,\cdots,p_n|P_1,\cdots,P_n[m(h,n,\{\phi_1,\cdots,\phi_n\})]) \ \simeq \ 1 \tag{3.70}$$

$n=2$ または $n=3$ のときでも，確率モデル $P(\vec{p}|h,n)$ を最尤推定することは難しい．なぜなら，複合助詞も含めると，一般に助詞 p_i の異り数は 200 を越え，また $\vec{p}$ を生成する確率モデルのパラメタ空間は n の数が増えるにつれて組み合わせ的に増大するからである．ところが，$\vec{p}$ に現われる全ての助詞が他の助詞と強く依存しているわけではなく，任意格のように他の助詞と独立に生成されると考えられる助詞も数多く存在する．そこで，他の助詞と依存関係があると思われる助詞の集合を P_d とし，確率モデルの後件 $\vec{p}$ に現われる助詞を P_d の要素のみに限定した．本実験では，EDR コーパスにおける出現頻度の高い助詞の中から以下の 17 個を人手で選択し，P_d の要素とした．

$$\begin{aligned} P_d \ = \ \{ & \text{を，は，に，が，で，も，から，では，の，には，と，} \\ & \text{でも，にも，まで，へ，より，とは} \} \end{aligned} \tag{3.71}$$

また，これ以外の助詞は常に他の助詞とは独立であると仮定し，その集合を $\overline{P_d}$ とする．そして，確率モデル $P(\vec{p}|v,n)$ を以下のように近似する．

$$P(\vec{p}|h,n) \simeq P(\vec{p'}|h,n) \cdot \prod_{p_i \in \overline{P_d}} P(p_i|PI) \tag{3.72}$$

ここで，$\vec{p'}$ は $\vec{p}$ の要素 p_i のうち $\overline{P_d}$ に属するものを PI に置き換えたものである．PI は $\overline{P_d}$ に属する助詞のいずれかが生成されることを表わす特別なシンボルである．また，$P(p_i|PI)$ は $\overline{P_d}$ に属する助詞の生成確率である．例えば,「昨日までに三万人が成田を出発した」という文において，3 個の助詞が動詞「出

発する」に係るという条件の下で，$\vec{p} = (\text{までに}, \text{が}, \text{を})$ が生成される確率を計算する場合を考える．このとき，助詞「までに」は $\overline{P_d}$ に属し，他の助詞「が」,「を」とは独立であるとみなして，3 つの助詞の生成確率を以下のように推定する．

$$P(\ (\text{までに}, \text{が}, \text{を}) \mid \text{出発する}, 3) \simeq P(\ (PI, \text{が}, \text{を}) \mid \text{出発する}, 3) \cdot P(\text{までに} \mid PI) \tag{3.73}$$

本実験においては，$P(p_i|PI)$ は表 3.3 に示した事例をもとに最尤推定した．一方，確率モデル $P(\vec{p'}|h, n)$ は以下のように推定した．

- 用言が動詞 v の場合

$$P(\vec{p'}|v, n) \ \simeq \ P(\vec{p'} \mid c_v, cf_v, syn_v, n) \tag{3.74}$$

式 (3.74) において，c_v は動詞 v の意味クラスである．本実験では，動詞の意味クラスとして分類語彙表の 3 桁の分類コードを利用した．さらに，動詞「ある」,「する」,「なる」については出現頻度も多く，また他の動詞とは異なる確率分布 $P(\vec{p'}|v, n)$ を持つと考えられるので，これらは特別に固有の意味クラスを持つとして取り扱った．実験に用いた動詞クラス c_v の異り数は 29 である．また，cf_v は動詞 v の格フレームを表わす．ここでいう格フレームとは，ある動詞 v が取り得る表層格を記載した語彙知識である．但し，動詞が取る表層格は，その動詞に係る助詞によって表わされるものとする．本実験では，表層格を表わす助詞の中でも「が」,「を」,「に」,「は」の 4 つに着目し，{ が, を, に, は } のうち動詞 v が表層格として取り得る助詞の部分集合を cf_v とした．したがって，cf_v の異り数は $2^4 = 16$ である．このような格フレームを日本語語彙体系に記載された文型パターンから作成した．ここでは，同じ意味クラス c_v に属する動詞，もしくは同じ格フレーム cf_v を持つ動詞は同じような確率分布 $P(\vec{p'}|v, n)$ を持つと仮定し，c_v, cf_v によって確率モデルの前件を近似している．一方，syn_v は動詞 v の統語的特性であり，次のいずれかの値を持つ．

  “連用” v が他の語を連用修飾するとき
  “連体” v が他の語を連体修飾するとき
  “主辞” v が何も修飾しないとき (文の最後に位置し，文全体の主辞となるとき)

syn_v を確率モデルの前件に加えたのは，確率分布 $P(\vec{p'}|v, n)$ がこのような統語的特性によって大きく影響されると考えられるからである．例えば，助詞「は」は文全体の主辞となる動詞に係りやすい傾向を強く持つ．以上のように，用言 h が動詞 v である場合には，v を意味クラス c_v，格フレーム cf_v，統語的特性 syn_v の組によって抽象化し，確率モデルを近似推定する．この確率モデル $P(\vec{p'} \mid c_v, cf_v, syn_v, n)$ の推定は最大エントロピー法を用いて行った．最大エントロピー法の詳細，および確率モデル推定の具体的な手順については 4 章で述べる．

- 用言 h が形容詞 a および名詞述語 np の場合

$$P(\vec{p'}|a, n) \ \simeq \ P(\vec{p'} \mid Adjective, syn_a, n) \tag{3.75}$$

$$P(\vec{p'}|np, n) \ \simeq \ P(\vec{p'} \mid Nominal_Predicate, syn_{np}, n) \tag{3.76}$$

式 (3.75),(3.76) において，syn_a, syn_{np} は syn_v と同じく形容詞，名詞述語の統語的特性を表わしている．また，$Adjective$, $Nominal_Predicate$ は，用言がそれぞれ形容詞，名詞述語であることを表わすシンボルである．用言が動詞のときのように意味クラスや格フレームを用いた抽象化を行わなかったのは，表 3.3 に示した通り，形容詞，名詞述語を用言 h とする訓練事例が動詞の場合に比べて少ないためである．すなわち，助詞とその係り先用言 a, np との間の語彙的従属関係を無視し，格間の従属関係のみを確率モデルに反映させている．これらの確率モデルは収集した訓練事例から最尤推定した．

3.2.2.3 体言に係る助詞に関する従属係数の学習

単語生成文脈決定規則 #4 によって，ある助詞 p の単語生成文脈が nd であると定められたときには，その従属係数 (3.77) が語彙モデルに反映される．

$$D(p|P[nd]) = \frac{P(p|P[nd])}{P(p|P)} \tag{3.77}$$

この従属係数を学習するために，EDR コーパスの全ての例文 (201,339 例文) から，体言に係る助詞 p をのべ 273,062 個収集した．式 (3.77) の分子はこの訓練データから最尤推定した．また，式 (3.77) の分母 $P(p|P)$ は，ここで収集した体言に係る助詞の事例と，表 3.3 に示した用言に係る助詞の事例から，同様に最尤推定した．尚，式 (3.66) の分母の各項 $P(p_i|P)$ は式 (3.77) の分母の確率モデルと同じものを使用した．

3.2.2.4 格要素以外の名詞の従属係数の学習

単語生成文脈決定規則 #5 によって，ある名詞 n の単語生成文脈が $(touten, mod_type)$ であると定められたときには，その従属係数 (3.78) が語彙モデルに反映される．

$$D(n|N[(touten, mod_type)]) = \frac{P(n|N[(touten, mod_type)])}{P(n|N)} \tag{3.78}$$

格要素に関する従属係数 $D(n|N[s(v,p)])$ を推定する際に，式 (3.58) のように名詞意味クラス c_n を用いて推定するパラメタ空間を縮小したのと同様に，式 (3.78) も名詞意味クラスを用いて以下のように近似した．

$$D(n|N[(touten, mod_type)]) \simeq \frac{\sum_j P(n|c_{n_j})P(c_{n_j}|N[(touten, mod_type)])}{P(n|N)} \tag{3.79}$$

名詞意味クラス c_{n_j} として，先ほどと同じ日本語語彙体系の名詞シソーラスにおけるルートから深さ 3 に位置する 151 個の意味クラスを利用した．

式 (3.79) 中の確率モデル $P(c_{n_j}|N[(touten, mod_type)])$ を学習するために，EDR コーパスの全ての例文 (201,339 例文) から，格要素以外の名詞とその $touten$(名詞が読点を伴っているか否か)，mod_type(名詞が連用修飾もしくは連体修飾しているか) の組で表現される事例 $(n, touten, mod_type)$ を収集した．また，これらの事例に含まれる名詞 n を名詞意味クラス c_n に抽象化し，$(c_n, touten, mod_type)$ といった事例の集合に変換した．ただし，名詞 n が複数の意味クラス $\{c_{n_1}, \cdots, c_{n_m}\}$ に属する場合は，事例 $(c_{n_j}, touten, mod_type)$ $(1 \leq j \leq m)$ の頻度を $1/m$ として数え上げた．そして，得られた事例の頻度 $O(c_n, touten, mod_type)$ から，式 (3.79) 中の確率モデル $P(c_{n_j}|N[(touten, mod_type)])$ を以下のように最尤推定した (簡単のため，$c_{n_j} = c_n$ としてある)．

$$P(c_n|N[(touten, mod_type)]) \simeq \frac{O(c_n, touten, mod_type)}{\sum_{c_n} O(c_n, touten, mod_type)} \tag{3.80}$$

また，式 (3.79) 中の残りの確率モデル $P(n|c_n)$ および $P(n|N)$ については，式 (3.58) の計算に用いた確率モデルと同じものを用いた．

3.2.3 実験結果

3.2.1 項で学習した構文モデル $P(R)$，および 3.2.2 項で学習した語彙モデル $P(W|R)$ を用いて，文節の係り受け解析の曖昧性解消実験を行った．

まず，テスト文として，京大コーパスの中から文節数 7〜9 の文をランダムに 500 文選び，これをテスト文とした．構文モデル $P(R)$ を学習する際に用いた訓練用例文にはこれらのテスト文は含まれていない．次に，テスト文の文節の係り受け解析を以下の手順により行った．

1. 入力文の各文節について，その品詞並びから文節ラベルを決定する．
2. 文節ラベルの列を入力として，テスト文の構文解析 (文節の係り受け解析) を行う．この構文解析には PGLR パーザを使用した．PGLR パーザは，文節ラベルの列を入力とし，係り受け解析結果の候補とその生成確率 $P(R)$ を出力する．
3. 2. で生成された係り受け解析結果の候補の中から，括弧に関する統語的制約に違反するものを除去した．この括弧に関する統語的制約を以下に示す．

   以下の文節列において，B_{open} を開括弧を含む文節，B_{close} を閉括弧を含む文節とする．

   $$A_1 \ \cdots \ A_n \ B_{open} \ B_1 \ \cdots \ B_m \ B_{close} \ C_1 \ \cdots \ C_l$$

   このとき，B_{open} の前にある文節 $A_1 \cdots A_n$ が文節 B_{open} および括弧の内部にある文節 $B_1 \cdots B_m$ に係ることはない．また，文節 B_{open} および $B_1 \cdots B_m$ が B_{close} の後にある文節 $C_1 \cdots C_l$ に係ることはない．

   例えば，以下の例文において，括弧の前に位置する文節 A_1(上田誠仁監督-が) が，文節 B_{open}(「-薄氷-を) や括弧の中に囲まれた文節 B_1(踏む)，B_2(気持ち-に) に係ることはないとみなす．また，文節 B_{open}(「-薄氷-を) や B_1(踏む)，B_2(気持ち-に) が，括弧の後に位置する文節 C_1(いう)，C_2(誤算-だっ-た) に係ることはないとみなす[6]．

A_1	B_{open}	B_1	B_2	B_{close}	C_1	C_2
上田誠仁監督-が	「-薄氷-を	踏む	気持ち-に	なっ-た-」-と	いう	誤算-だっ-た

4. 各解析結果の候補に対して，それぞれの構文モデル $P(R)$ の値に語彙モデル $P(W|R)$ の値をかける．3.2.2 項の冒頭で述べた通り，本実験においては，語彙モデル $P(W|R) = P_{cf}(W|L) \cdot D(W|R)$ のうち，$P_{cf}(W|L)$ の計算は省略できるので，従属係数の積 $D(W|R)$ のみを計算する．

文節数が 7〜9 という比較的文長の短い例文をテスト文として選んだのは，2. で構文解析を行う PGLR パーザがまだ開発の途中であり，長い文長の例文の係り受け解析に非常に多くの時間を要するためである．

次に，このようにして得られた係り受け解析結果の評価を行った．ここでは，評価尺度として以下の 2 つを用いた．

- 文節の正解率

$$\text{文節の正解率} \ = \ \frac{\text{係り先の正しい文節の数}}{\text{テスト文に含まれる文節の数}} \tag{3.81}$$

この文節の正解率はそれぞれの例文の生成確率が一位である解析結果の候補について計算する．但し，文の最後に位置する 2 つの文節は評価の対象から除外する．これは，文の一番最後にある文節は係り先がなく (文全体の主辞となっている)，また文の最後から 2 番目にある文節の係り先は常に文の一番最後に位置する文節であるからである．

[6] 文節が開括弧，閉括弧を持つという情報を文節ラベルに反映させ，このような括弧に関する統語的制約を CFG 規則によって具現化する方法も考えられるが，CFG 規則の数および構文モデルとして用いる PGLR の推定パラメタ数が増大するといった問題がある．そこで本研究では，係り受け解析を行う際には括弧に関する制約を考慮せず，括弧に関する統語的制約に違反する解析結果の候補を後処理によって削除するというアプローチを取る．

- 文の正解率

$$\text{文の正解率} = \frac{\text{生成確率の上位 } k \text{ 位の候補の中に正しい解析結果が含まれている文の数}}{\text{テスト文の総数}} \tag{3.82}$$

正しい解析結果とは，例文中の全ての文節の係り先が正しい解析結果の候補を指す．

まず，解析結果の候補に与えるスコアとして，構文モデル $P(R)$ のみによる生成確率を与えたときの文節の正解率，および $k = 1, 5, 10$ のときの文の正解率を表 3.4 に示す．

表 3.4: 構文モデルのみを用いた場合の解析結果

	文節の正解率	文の正解率		
		$k = 1$	$k = 5$	$k = 10$
ベースライン	61.68%	0.6%	—	—
$P(R)$	73.38%	15.8%	47.0%	61.2%

表 3.4 におけるベースラインとは，(1) 全ての文節は係り得るなるべく近い文節に係る，(2) 一文中における文節の係り受け関係は互いに交差しない，として文節の係り先を決定したときの解析結果を表わす．構文モデルによる生成確率を解析結果のスコアとして用いたとき，このベースラインに比べて，文節の正解率が 11.7%，文の正解率 ($k = 1$) が 15.2% 向上することがわかった．

3.2.1.3 で述べたように，構文モデルは京大コーパス 9,944 文から学習した．この訓練データの量を 1/4, 1/2，3/4 と変化させて構文モデルを学習したときの文節の正解率，文の正解率の変化を図 3.13 に示す．

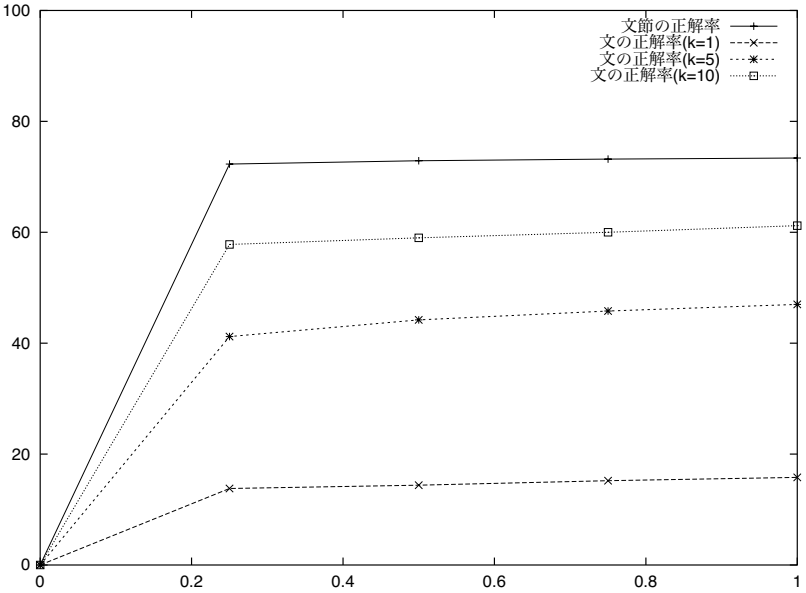


図 3.13: 学習データ量を変化させたときの正解率の変化

この図から，文節の正解率，文の正解率ともに飽和傾向にあることがわかる．3.1.1 項で述べた通り，構文モデルの学習には比較的作成コストの高い構文構造付きコーパスが必要である．しかしながら，図 3.13 の結果から，現時点で利用可能な構文構造付きコーパス (本実験においては京大コーパス) を訓練データとしても，構文モデルを十分に学習することが可能であるといえる．

次に，構文モデル $P(R)$ と語彙モデル $P(W|R)$ の積 $P(R) \cdot P(W|R)$ を解析結果のスコアとしたときの文節の正解率，および $k = 1, 5, 10$ のときの文の正解率を表 3.5 に示す．3.2.2 項で述べたように，今回の実験で用いた語彙モデルにおいてはいくつかの種類の統計情報を取り扱う．ここでは，それぞれの統計情報の曖昧性解消における効果を調べるために，以下に述べる 6 種類の語彙モデルを用意し，それらを比較した．

F: 格要素となる名詞に関する従属係数のみを用いたモデル

$D(W|R)$ として，式 (3.55) によって与えられる従属係数のみを考慮したモデルである．

Md: 用言に係る助詞に関する従属係数のみを用いたモデル

$D(W|R)$ として，式 (3.67) によって与えられる従属係数のみを考慮したモデルである．

Mi: 用言に係る助詞に関する従属係数のみを用いたモデル (格独立)

“Md” と同様に用言に係る助詞に関する従属関係を考えたモデルである．モデル “Md” では助詞とその係り先用言の間の語彙的従属関係，および格間の従属関係の両方を考慮に入れていたが，ここでは格間の従属関係を無視する．すなわち，同じ用言に係る助詞は互いに独立に生成されると仮定する．このモデルは，式 (3.67) を以下のように計算することによって実現される．

$$
\begin{aligned}
& D(p_1, \cdots, p_n | P_1, \cdots, P_n[m(h, n, \{\phi_1, \cdots, \phi_n\})]) && (3.83) \\
\simeq \quad & \prod_i D(p_i | P_i[h]) && (3.84) \\
= \quad & \prod_i \frac{P(p_i | P_i[h])}{P(p_i | P_i)} && (3.85)
\end{aligned}
$$

式 (3.85) における $P(p_i|P_i[h])$ は，ある助詞 P_i が用言 h に係るときに，単語 p_i が生成される確率を表わしている．

P: 体言に係る助詞に関する従属係数のみを用いたモデル

$D(W|R)$ として，式 (3.77) によって与えられる従属係数のみを考慮したモデルである．

N: 格要素以外の名詞に関する従属係数のみを用いたモデル

$D(W|R)$ として，式 (3.78) によって与えられる従属係数のみを考慮したモデルである．

all: 全ての従属係数を用いたモデル

上記全ての従属係数を考慮したモデルである．但し，用言に係る助詞に関しては，助詞とその係り先用言の間の語彙的従属関係と格間の従属関係の両方を考慮する．

表 3.5 の “正解率の差” の欄は，それぞれのモデルの文節の正解率，文の正解率が構文モデルのみを用いたときと比べてどれだけ向上したかを示している．この表から，語彙モデルで考慮した様々な種類の語彙的従属関係のうち，体言に係る助詞に関する従属係数が正解率の向上に一番大きく貢献することがわかる．すなわち，その助詞が用言に係るか体言に係るかの違いが助詞の生成確率に大きく影響し，その違いを考慮することによって曖昧性解消の精度を大きく向上させることができた．また，モデル “Mi” と “Md” との

表 3.5: 構文モデルと語彙モデルを利用した場合の解析結果

	文節の正解率	正解率の差
$P(R)$	73.38%	—
$P(R) \cdot P(W\|R)$ (F)	74.69%	+1.31
$P(R) \cdot P(W\|R)$ (Md)	78.55%	+5.17
$P(R) \cdot P(W\|R)$ (Mi)	74.92%	+1.54
$P(R) \cdot P(W\|R)$ (P)	82.22%	+8.84
$P(R) \cdot P(W\|R)$ (N)	73.61%	+0.23
$P(R) \cdot P(W\|R)$ (all)	84.34%	+10.96

	文の正解率 ($k = 1$)	正解率の差
$P(R)$	15.8%	—
$P(R) \cdot P(W\|R)$ (F)	16.2%	+0.4
$P(R) \cdot P(W\|R)$ (Md)	25.4%	+9.6
$P(R) \cdot P(W\|R)$ (Mi)	20.2%	+4.4
$P(R) \cdot P(W\|R)$ (P)	34.4%	+18.2
$P(R) \cdot P(W\|R)$ (N)	16.2%	+0.4
$P(R) \cdot P(W\|R)$ (all)	39.4%	+23.6

	文の正解率 ($k = 5$)	正解率の差
$P(R)$	47.0%	—
$P(R) \cdot P(W\|R)$ (F)	50.4%	+3.4
$P(R) \cdot P(W\|R)$ (Md)	59.2%	+12.2
$P(R) \cdot P(W\|R)$ (Mi)	50.8%	+3.8
$P(R) \cdot P(W\|R)$ (P)	63.4%	+16.4
$P(R) \cdot P(W\|R)$ (N)	48.8%	+1.8
$P(R) \cdot P(W\|R)$ (all)	73.8%	+26.8

	文の正解率 ($k = 10$)	正解率の差
$P(R)$	61.2%	—
$P(R) \cdot P(W\|R)$ (F)	63.2%	+2.0
$P(R) \cdot P(W\|R)$ (Md)	72.0%	+10.8
$P(R) \cdot P(W\|R)$ (Mi)	65.8%	+4.6
$P(R) \cdot P(W\|R)$ (J)	75.2%	+14.0
$P(R) \cdot P(W\|R)$ (N)	63.0%	+1.8
$P(R) \cdot P(W\|R)$ (all)	82.0%	+20.8

比較により，助詞とその係り先用言との間の語彙的従属関係だけでなく，格間の従属関係もまた曖昧性解消に有効に働くと考えられる．

構文モデルと全ての語彙的従属関係を考慮した語彙モデルを組み合わせた確率を曖昧性解消のためのスコアとして用いることにより (モデル “all”)，構文モデルのみを用いたときと比べて，文節の正解率で 10.96%，$k = 1, 5, 10$ のときの文の正解率でそれぞれ 23.6%，26.8%，20.8% 向上することがわかった．構文モデルのみを曖昧性解消に用いたときのベースラインとの文節の正解率，および文の正解率 ($k = 1$) の差がそれぞれ 11.7%，15.2%であることから (表 3.4)，曖昧性解消の精度向上において，語彙モデルは構文モデルと同程度の貢献をしていると考えられる．本研究で提案した統合的確率言語モデルにおいては，過去の多くの先行研究と異なり，語彙的な統計情報を局所化し構文的な統計情報と独立に学習している．このようなアプローチにおいても，語彙的な統計情報は曖昧性解消の精度向上に十分大きく貢献すると期待できる．

本実験における語彙モデルで考慮した語彙的従属関係はある特定のタイプの文節の係り先の決定に大きく影響する．すなわち，表 3.5 におけるモデル “F”，“Md”，“Mi”，“P” は文節タイプが “後置詞文節” である文節の係り先に大きく影響し，モデル “N” は文節タイプが “名詞” である文節の係り先に大きく影響する．そこで，それぞれの語彙的従属関係を考慮したモデルにおいて，これらの文節のみを対象とした文節の正解率を調べた．結果を表 3.6 に示す．

表 3.6: 文節タイプ毎の正解率

	文節タイプ	正解文節数	文節総数	文節の正解率	正解率の差
$P(R)$	後置詞文節	1,236	1,772	69.75%	—
$P(R) \cdot P(W\|R)$ (F)	後置詞文節	1,266	1,772	71.44%	+1.69
$P(R) \cdot P(W\|R)$ (Md)	後置詞文節	1,392	1,772	78.56%	+8.81
$P(R) \cdot P(W\|R)$ (Mi)	後置詞文節	1,285	1,772	72.52%	+2.77
$P(R) \cdot P(W\|R)$ (P)	後置詞文節	1,493	1,772	84.26%	+15.51
$P(R)$	名詞	217	315	68.89%	—
$P(R) \cdot P(W\|R)$ (M)	名詞	242	315	76.83%	+7.94

表 3.5, 表 3.6 から，それぞれの語彙的従属関係が強く影響する文節タイプのみで評価したときには，全ての文節タイプで評価したときと比べて，構文モデルのみを用いた場合との正解率の差が大きくなっていることがわかる．特に，名詞の従属係数のみを考慮したモデル “N” の場合，表 3.5 では 0.23%しか正解率が向上していないが，これは文節タイプが “名詞” である文節の数が少ないためであり，これらの文節のみを評価対象としたときの文節の正解率は 7.94%向上していることがわかる．

最後に，本研究で提案する統合的確率言語モデルを用いた解析結果と KNP パーザ [46] による解析結果との比較を行った．KNP パーザは文節の素性を決定する規則を人手で作成し，その素性をもとに文節間の係り受け解析を行う．また，文中に並列構造が含まれる可能性があればそれを認識し，並列構造内の文節の類似性を係り受け解析の手がかりとすることに特徴がある．KNP パーザは形態素解析システム JUMAN[60] の形態素解析結果を入力とし，文節区切りを認定してから文節の係り受け解析を行う．そこで，本項の実験で用いた 500 個のテスト文のうち，JUMAN の形態素解析結果による形態素区切りおよび KNP パーザによる文節区切りの結果がコーパスと一致した 388 文を対象に，両者の係り受け解析結果の比較を行った．結果を表 3.7 に示す．

本手法は KNP パーザよりも文節の正解率で 1%程度劣っている．しかしながら，実験で用いた語彙モデ

表 3.7: KNP パーザとの比較

	文節タイプ	正解文節数	文節総数	文節の正解率
本手法	全て	1,940	2,295	84.53%
KNP パーザ	全て	1,973	2,302	85.71%
本手法	後置詞文節	1,186	1,370	86.57%
KNP パーザ	後置詞文節	1,189	1,370	86.79%
本手法	名詞	153	199	76.88%
KNP パーザ	名詞	160	199	80.40%

ルがその係り受け先の決定に強く影響する “後置詞文節” や “名詞” のみで評価した場合には，両者の正解文節数はほぼ等しい．また，3.2.4.3 で述べるように，曖昧性解消に有効ではあるが今回の実験では考慮されていない語彙的従属関係も数多く存在する．このような曖昧性解消に有効な語彙的従属関係を新たに統合的確率言語モデルに組み込むことにより，KNP パーザと同等またはそれ以上の精度で文節の係り受け解析を行うことができると期待される．

3.2.4 考察

前項の実験においては，構文的優先度や語彙的従属関係などの統計情報が曖昧性解消に有効であることを確認できたものの，解析精度そのものは十分に高いとはいえず，改善の余地がある．そこで，文節の係り受け解析に失敗した事例を分析し，その原因を調査した．本項では，曖昧性解消に失敗する主な原因と，本研究で提案する統合的確率言語モデルにおけるそれらの対処法について述べる．

まず，テスト文における文節タイプの分布，および文節タイプ別にみた誤り率を調査した．表 3.8 は，文節タイプ毎の文節総数，係り先を誤った文節数，および文節の誤り率を示したものである．但し，構文モデル $P(R)$ と全ての語彙的従属関係を考慮した語彙モデル $P(W|R)$ の両方を用いた確率モデル (表 3.5 におけるモデル “all”) によって 1 位の生成確率が与えられる解析結果の候補について調査を行った．

表 3.8: 文節タイプ別の誤り率

文節タイプ	誤り文節数	文節数	文節の誤り率
後置詞文節	245	1,788	13.70%
名詞	72	315	22.86%
動詞，形容詞，名詞述語	108	683	15.81%
副詞，感動詞など	33	145	22.76%
その他	8	44	18.18%
合計	466	2,975	15.66%

表 3.8 において，文節タイプが “動詞，形容詞，名詞述語” となっているのは受け属性として “格” を含む文節を示しており，“副詞，感動詞など” となっているのは受け属性として “連用” のみを持つ文節の誤り

率を示している (文節の受け属性については表 3.1 を参照).

今回の実験で使用した語彙モデルに反映されている語彙的従属関係は主に “後置詞文節” の係り先に強く影響する．そのため，文節タイプが “後置詞文節” である文節の誤り率は全体に比べて低くなっている．しかしながら，“後置詞文節” の文節全体に占める割合は約 60%と大きく，誤り文節の数においても，466 文節のうち約半分の 245 文節が “後置詞文節” となっている．したがって，文節の正解率をさらに向上させるには，“後置詞文節” の係り先を誤った原因を調査し，それに基づきモデルを改善する必要がある．“後置詞文節” の係り先を誤る主な原因については 3.2.4.1，3.2.4.2 で述べる．

これに対し，“後置詞文節” 以外の文節の係り先を決定する際には，今回の実験で使用した語彙モデルにおいては語彙的従属関係は考慮されていない．また，文節タイプが “名詞” の文節の係り先については式 (3.78) に示した従属係数が考慮されるが，3.2.4.3 で述べるように，文節タイプが “名詞” である文節の係り先を決める際に考慮すべき統計情報はこれだけではない．そのため，“後置詞文節” 以外の文節タイプの文節の誤り率は全ての文節タイプの文節誤り率と比べて同程度もしくは高くなっていることがわかる．したがって，今回の実験で考慮されていない語彙的従属関係を新たに統合的確率言語モデルに組み込むことにより，文節の正解率を向上させることができると考えられる．これについては 3.2.4.3 で詳しく述べる．

3.2.4.1 並列構造

入力文に並列構造が含まれている場合には，文節タイプが “後置詞文節” である文節の係り先を誤る場合が多い．並列構造を含む入力文，およびその解析結果の候補の例を図 3.14 に示す．

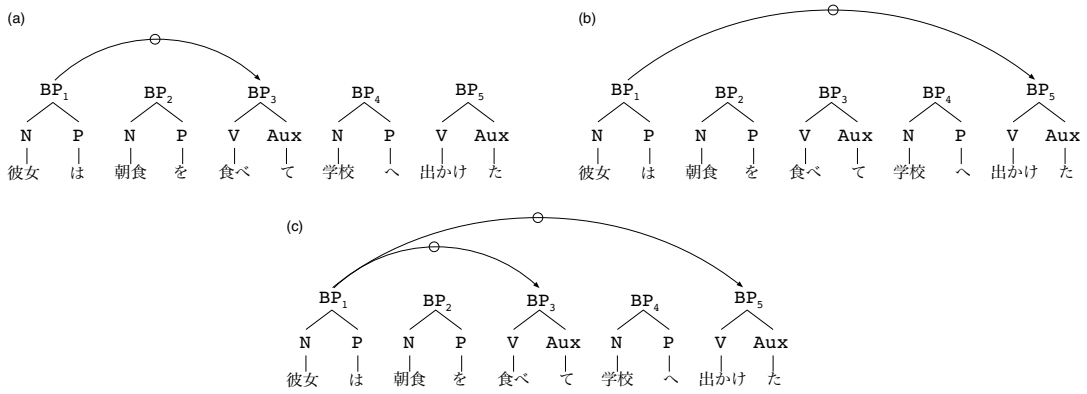


図 3.14: 並列構造を含む例文

図 3.14 の例文において，名詞「彼女」は動詞「食べ」のハ格の格要素でもあり，動詞「出かけ」のハ格の格要素でもある．このとき，図 3.14 の (c) のように，文節「彼女-は」は 2 つの文節「食べ-て」と「出かけ-て」の両方に係ると解釈した方が自然である．ところが，3.2.3 項の実験では，文節「彼女-は」が (a)「食べ-た」のみに係るか，(b)「出かけ-た」のみに係るかといった 2 つの解釈のどちらかを必ず選択していた．ところが，並列構造を含む文において，解釈 (a) と (b) のどちらが正しいかを選択するのはあまり意味のないことである．

これに加えて，本研究における統合的確率言語モデルは，このような場合に解釈 (a) に対して高い生成確率を与える傾向を持つ．語彙モデル $P(W|R)$ においては，2 つの解釈 (a),(b) に対してそれぞれ式 (3.86),

式 (3.87) に示す格要素に関する従属係数が考慮される．

$$\text{(a)} \quad D(\text{彼女}\,|N[s(\text{食べ}, \text{は})]) \tag{3.86}$$

$$\text{(b)} \quad D(\text{彼女}\,|N[s(\text{出かけ}, \text{は})]) \tag{3.87}$$

この 2 つの従属係数はともに 1 より大きな値をとると予想されるので，2 つの解釈 (a),(b) に語彙モデル $P(W|R)$ が与える生成確率には大きな差はない．なぜなら,「彼女」という単語は，動詞「食べ」のハ格の格要素としても現われやすく，また動詞「出かけ」のハ格の格要素としても現われやすいからである．一方構文モデル $P(R)$ においては，より近くの文節に係るという解釈が優先されるため，解釈 (a) に高い確率が割り当てられる．ところが，今回の実験でテスト文として用いた京大コーパスにおいては，図 3.14 のような並列構造がある場合，文節「彼女-は」の係り先は (b) のような一番遠くの文節となっているため[7]，結果として文節「彼女-は」の係り先の認定に失敗したとみなされる．このように，入力文に並列構造が含まれている場合を考慮していないことが，文節タイプが “後置詞文節” である文節の係り先を誤る主な原因のひとつとなっている．

このような並列構造に対処するためには，文節「彼女-は」が 2 つの文節「食べ-た」と「出かけ-た」の両方に係るという解釈 (c) を導入し，3 つの解釈 (a),(b),(c) の中から正しい解釈を選択するようにシステムを拡張する．そして，解釈 (c) においては,「彼女」は動詞「食べ」のハ格の格要素でもあり動詞「出かけ」のハ格の格要素でもあるので，3.1.2.3 で述べたように「彼女」が複数の単語生成文脈を持つとみなし，それぞれの単語生成文脈に対応する従属係数の積を与えればよい．

$$\text{(c)} \quad D(\text{彼女}\,|N[s(\text{食べ}, \text{は})]) \cdot D(\text{彼女}\,|N[s(\text{出かけ}, \text{は})]) \tag{3.88}$$

この 2 つの従属係数 $D(\text{彼女}\,|N[s(\text{食べ}, \text{は})])$, $D(\text{彼女}\,|N[s(\text{出かけ}, \text{は})])$ の値はともに 1 より大きくなると考えられるので，式 (3.88) の値は式 (3.86) や式 (3.87) の値より十分大きくなり，結果として正しい解釈 (c) に高い生成確率が与えられると期待できる．

3.2.4.2 補助用言

文節タイプが “後置詞文節” である文節の係り先を間違えるもうひとつの大きな原因として，単語生成文脈決定規則によって決められた単語生成文脈 c が不適切な場合があることが挙げられる．このことを図 3.15 を例に説明する．

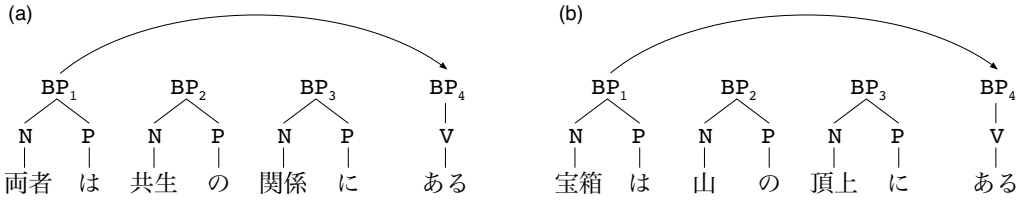


図 3.15: 補助動詞「ある」を含む例文

図 3.15 の例文 (a),(b) において，最初の文節「両者-は」と「宝箱-は」はともに最後の文節「ある」に係っている．すなわち,「両者」と「宝箱」はともに動詞「ある」のハ格の格要素となっている．したがって，単

[7] 京大コーパスにも並列に関する情報は付加されているが，これは本研究で考える並列の基準とは若干ずれているため，今回の実験ではこの情報は用いなかった．

語生成文脈決定規則 #2 により，それぞれの単語生成文脈はともに s(ある, は) であると定められ，以下に示す従属係数が考慮される．

$$\text{(a)}\quad D(\text{両者}\,|N[s(\text{ある}, \text{は})]) \tag{3.89}$$

$$\text{(b)}\quad D(\text{宝箱}\,|N[s(\text{ある}, \text{は})]) \tag{3.90}$$

ところが，例文 (a) においては，式 (3.89) の従属係数は適切であるとは言えない．なぜなら，例文 (a) における動詞「ある」は補助動詞として働き，例文 (b) のように「ある」本来が持つ “存在する” という意味で用いられているわけではないからである．むしろ,「両者」は動詞「ある」だけではなく,「関係にある」という句全体と強い従属関係を持つ．したがって,「両者」の単語生成文脈は s(ある, は) ではなく s(関係にある, は) とすべきであり，式 (3.89) の代わりに以下のような従属係数を考慮すべきである．

$$D(\text{両者}\,|N[s(\text{関係にある}, \text{は})]) \tag{3.91}$$

この従属係数は,「N が関係にある」という解釈が与えられたときに，その名詞 N として「両者」という単語がどの程度現われやすいかを示した統計量である．このように動詞が補助的な意味しか持たないときには，単語生成文脈決定規則 #2 により決定された単語生成文脈が不適切な場合がある．

補助的な意味しか持たず，そのため不適切な単語生成文脈が選ばれる可能性のある動詞は「ある」だけではない．図 3.16 にその例を挙げる．

(a) 対策-が / 必要-と / なる
× $D(\text{対策}\,|N[s(\text{なる}, \text{が})])$
○ $D(\text{対策}\,|N[s(\text{必要となる}, \text{が})])$

(b) 太郎-が / 英語-の / 勉強-を / する
× $D(\text{太郎}\,|N[s(\text{する}, \text{が})])$
○ $D(\text{太郎}\,|N[s(\text{勉強する}, \text{が})])$

(c) 締切-が / 近づい-て / いる
× 締切: $D(\text{締切}\,|N[s(\text{いる}, \text{が})])$
○ 締切: $D(\text{締切}\,|N[s(\text{近づく}, \text{が})])$

図 3.16: 不適切な単語生成文脈が選ばれる例文

図 3.16 の各例文において，“×” は単語生成文脈決定規則 #2 によって定められる不適切な単語生成文脈を，“○” は考慮すべき適切な単語生成文脈を表わす．例文 (a) において，文節「対策-が」が「なる」に係るとき，単語生成文脈決定規則 #2 により，名詞「対策」の従属係数は $D(\text{対策}\,|N[s(\text{なる}, \text{が})])$ となる．しかしながら，この例文における「なる」は補助的な意味しか持たないので，これは $D(\text{対策}\,|N[s(\text{必要となる}, \text{が})])$ とすべきである．また，例文 (b) において，文節「太郎-が」が「する」に係っている場合を考える．このときもまた，動詞「する」は補助的な意味しか持たないので，名詞「太郎」の従属係数は $D(\text{太郎}\,|N[s(\text{する}, \text{が})])$ ではなく $D(\text{太郎}\,|N[s(\text{勉強する}, \text{が})])$ とすべきである.「いる」が本来の意味である “存在する” という意味を持たない例文 (c) においても同様のことが言える．名詞「締切」の従属係数は $D(\text{締切}\,|N[s(\text{いる}, \text{が})])$ ではなく，$D(\text{締切}\,|N[s(\text{近づく}, \text{が})])$ とすべきである．

以上で述べたことは全て単語生成文脈決定規則を洗練することにより対処できる．すなわち,「ある」,「なる」,「する」,「いる」などの補助的な意味しか持たない可能性のある動詞については，適切な単語生成文脈が選択されるように例外処理を行えばよい．本研究では，単語生成文脈決定規則は人手で記述することを前提にしているので，このような例外処理に対する対応は比較的柔軟に行うことができる．

補助動詞に関して例外処理を行うように単語生成文脈決定規則を洗練することは，格要素間の従属関係を考慮しているとも考えられる．例えば，図 3.15 の例文 (a) においては式 (3.91) のような従属係数を考えるべきだと述べたが，これは同じ動詞「ある」の格要素「両者」と「関係」の間の従属関係を確率モデルに反映させていると捉えることもできる．同様に図 3.16 の例文においても，(a) 動詞「なる」の格要素となっている「対策」と「必要」の従属関係，(b) 動詞「する」の格要素となっている「太郎」と「勉強」の従属関係を考慮していると考えられる．

このような格要素間の従属関係は，動詞が補助動詞である場合にのみ存在するわけではない．例えば，以下の例文 1. は自然だが，例文 2. は不自然である．

1. 牛が草を食べる
2. 牛がステーキを食べる

ここで注意しなければならないのは,「ステーキ」は「食べる」のヲ格の格要素として現われにくいわけではないということである．すなわち,「食べる」のガ格の格要素が「牛」であれば「ステーキ」は「食べる」のヲ格の格要素として現れにくいが,「食べる」のガ格の格要素が「人間」であればそうではない．これは,「食べる」という動詞のガ格とヲ格の格要素の間にも語彙的従属関係が存在することを意味する．このような格要素間の従属関係を考慮するには，以下のような従属係数を学習すればよい．

$$D(\text{牛}\,|N[\text{食べる}, \{\,\text{が}, \phi\,\}, \{\,\text{を}, \text{草}\,\}]) \quad = \quad \frac{P(\text{牛}\,|N[\text{食べる}, \{\,\text{が}, \phi\,\}, \{\,\text{を}, \text{草}\,\}])}{P(\text{牛}\,|N)} \tag{3.92}$$

式 (3.92) の分子は，ある名詞 N が「食べる」のガ格の格要素でありかつ「食べる」のヲ格の格要素が「草」であるときに，その名詞から「牛」が生成される確率である．この確率モデルには,「牛」と「食べる」といった格要素とその係り先となる動詞の間の従属関係や,「牛」と「草」といった格要素間の従属関係が反映されている．しかしながら，このような格要素間の従属関係を確率モデルに反映させるべきではあるのだが，式 (3.92) のような従属係数はパラメタ数が多く，その推定は困難である．したがって，全ての動詞について格要素間の従属関係を考慮するのではなく，動詞が「ある」,「なる」などの補助動詞の場合に限り格要素間の従属関係を考慮し，式 (3.91) のような従属係数を確率モデルに加えることが，現実的な対応策として考えられる．

3.2.4.3 実験で考慮しなかった語彙的従属関係

本実験で用いた語彙モデルにおいては，動詞・助詞の組とその格要素となる名詞との間の語彙的従属関係，助詞とその係り先用言との間の語彙的従属関係，格間の従属関係などを確率モデルに反映させている．しかしながら，曖昧性解消に有効であると考えられるこれら以外の語彙的従属関係も存在する．ここでは，そのような語彙的従属関係の例を 3 つ挙げる．

1. 名詞の連体修飾に関する語彙的従属関係
   例えば,「彼女の紫色の帽子が風に飛ばされた」という文においては，図 3.17 に示すように，文節「彼女-の」が「紫色-の」に係るという解釈 (a) と「帽子-が」に係るという解釈 (b) の 2 つの解釈がある．正しい解釈は文節「彼女-の」が「帽子-が」に係る解釈 (b) である．しかしながら，今回の実験で学習した語彙モデルにおいては,「彼女」が「帽子」に連体修飾することはあっても,「紫色」に連体修飾することは少ないといった，名詞の連体修飾に関する語彙的従属関係は考慮されていない．一方，構文モデルにおいては，近い文節に係りやすいといった距離に関する優先度が反映されているために解釈 (a) に対して高い確率を与える．結果として，解釈 (a) の生成確率は解釈 (b) の生成確率よりも大きくなり，文節「彼女-の」の係り先の認定に失敗する．以上の例では文節タイプが “後置詞文節” である文

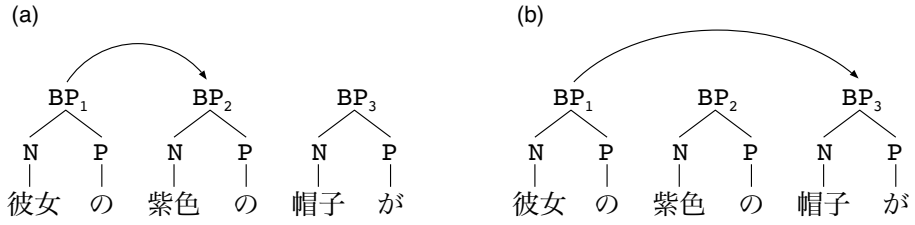


図 3.17: 名詞の連体修飾先の曖昧性

節が連体修飾する場合を考えていたが，文節タイプが “名詞” である文節が連体修飾する場合にも同様の問題が生じる．

これを回避するためには，以下のような従属係数を学習し，語彙モデルに加えればよい．

$$D(n_1|N[n_2]) = \frac{P(n_1|N[n_2])}{P(n_1|N)} \tag{3.93}$$

式 (3.93) の分子 $P(n_1|N[n_2])$ は，ある名詞 N が n_2 を連体修飾しているとき，その名詞として単語 n_1 が生成される確率を表わしている．したがって，n_1 が n_2 を連体修飾しやすければしやすいほど，$D(n_1|N[n_2])$ の値は 1 より大きくなると予想される．例えば，図 3.17 における 2 つの解釈 (a),(b) においては，以下のような従属係数が考慮される．

$$\text{(a)} \quad D(\text{彼女}\,|N[\text{紫色}]) \tag{3.94}$$

$$\text{(b)} \quad D(\text{彼女}\,|N[\text{帽子}]) \tag{3.95}$$

これらの従属係数を訓練データから適切に学習することができれば，$D(\text{彼女}\,|N[\text{帽子}])$ の値は $D(\text{彼女}\,|N[\text{紫色}])$ よりも十分大きくなり，正しい解釈 (b) に高い生成確率を与えると期待できる．

名詞の連体修飾に関する語彙的従属関係を考慮する際には，名詞の連体修飾には様々な種類がある [87] ということに注意しなければならない．例えば，係り側の名詞が係り先の名詞の所有者であることを表わす場合や，係り側の名詞が係り先の名詞を限定している場合などがある．

- 彼女の本　(「本」は「彼女」の所有物である)
- 講談社の本　(「講談社」は「本」を限定している)

連体修飾関係の種類によって，従属係数 (3.93) の分布も異なると考えられる．したがって，式 (3.96) に示すように，従属係数の単語生成文脈に連体修飾の関係の種類 rel を加えるべきである．

$$D(n_1|N[n_2, rel]) \tag{3.96}$$

しかしながら，このような連体修飾の関係の認定は一般に簡単ではないといった問題点もある．

2. 動詞の連体修飾に関する語彙的従属関係

動詞が連体修飾するときもまた同様の問題が生じる．例えば,「昨日買った数学の本がなくなった」という文においては，図 3.18 に示すように，文節「買っ-た」が「数学-の」に係るという解釈 (a) と「本-が」に係るという解釈 (b) の 2 通りの解釈がある．

正しい解釈は文節「買っ-た」が「本-が」に係る解釈 (b) であるが，先ほどと同様の理由で，近い文節「数学-の」に係る解釈 (a) に高い確率が与えられ，文節「買っ-た」の係り先の認定に失敗する．

これを回避するためには，以下のような従属係数を考慮すればよい．

$$D(n|N[v]) = \frac{P(n|N[v])}{P(n|N)} \tag{3.97}$$

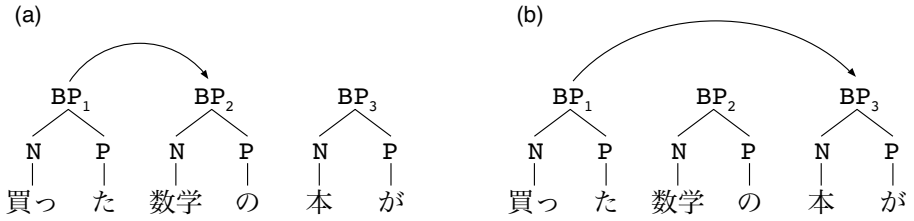


図 3.18: 動詞の連体修飾先の曖昧性

式 (3.97) の分子 $P(n|N[v])$ は，ある名詞 N が動詞 v によって連体修飾されているとき，その名詞として単語 n が生成される確率を表わしている．例えば，図 3.18 における 2 つの解釈 (a),(b) においては，以下のような従属係数が考慮される．

$$\text{(a)} \quad D(\text{数学}\,|N[\text{買っ}]) \tag{3.98}$$

$$\text{(b)} \quad D(\text{本}\,|N[\text{買っ}]) \tag{3.99}$$

この場合，連体修飾を受ける名詞は動詞「買っ」のヲ格の格要素となっている．したがって,「数学」よりも「本」の方が動詞「買っ」のヲ格として現われやすいので，式 (3.97) の従属係数を適切に学習することができれば，$D(\text{本}\,|N[\text{買う}])$ の値は $D(\text{数学}\,|N[\text{買う}])$ よりも十分大きくなり，正しい解釈 (b) に高い確率を与えると期待できる．

動詞とその連体修飾先となる名詞の間の語彙的従属関係は，式 (3.97) のように名詞の生成確率に反映させる代わりに，式 (3.100) のように動詞の生成確率に反映させることもできる．

$$D(v|V[n]) = \frac{P(v|V[n])}{P(v|V)} \tag{3.100}$$

$P(v|V[n])$ はある動詞 V が名詞 n を連体修飾するときに，その動詞として単語 v が生成される確率である．しかしながら，連体修飾を受ける名詞が動詞のヲ格の格要素であるとわかっている場合には，式 (3.97) は式 (3.55) の p を「を」としたときとほぼ等しいと考えられる．

$$D(n|N[v]) \;\simeq\; D(n|N[s(v,\text{を})]) \tag{3.101}$$

これは式 (3.97) の動詞の連体修飾に関する従属係数を，“本を買う” といった格関係を表わす事例からも学習できることを意味する．したがって，動詞とその連体修飾先となる名詞の間の語彙的従属関係を学習する際には，式 (3.100) よりも式 (3.97) のような従属係数を用いた方が，より多くの学習データを利用することができる．

ここで問題となるのは，連体修飾を受ける名詞が必ずしも動詞のヲ格の格要素となるわけではないということである．例えば，“燃えた本” の場合，名詞「本」は動詞「燃える」のガ格の格要素となっている．また，“考えた行動” の場合，両者の関係は格関係ではなく，動詞「考えた」は名詞「行動」の意味を限定している．したがって，先ほどの名詞の連体修飾の場合と同様に，動詞の連体修飾についてもその修飾関係の種類を区別する必要がある [3, 4].

3. 副詞の係り先

副詞とその係り先となる用言との間にも語彙的従属関係は存在する．例えば,「決っして妥協したわけではない」という文においては，図 3.19 に示すように，文節「決っして」が「妥協-した」に係るという解釈 (a) と「ない」に係るという解釈 (b) の 2 つの解釈がある．

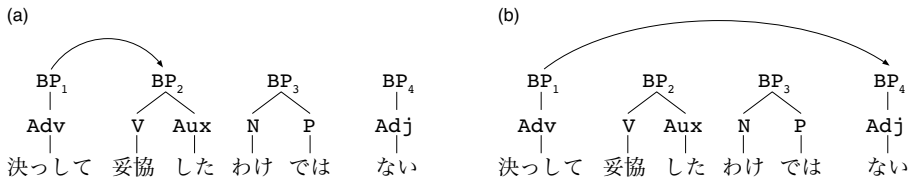


図 3.19: 副詞の係り先の曖昧性

今回の実験で学習した語彙モデルでは，副詞と係り先の用言との間の語彙的従属関係は考慮されていない．したがって，語彙モデルは解釈 (a),(b) ともに等しい確率を与える．これに対して，構文モデルにおいては近い文節に係る解釈 (a) の確率が高くなり，結果として文節「決っして」の係り先の認定に失敗する．
これを回避するためには，以下のような従属係数を考慮すればよい．

$$D(adv|ADV[h]) = \frac{P(adv|ADV[h])}{P(adv|ADV)} \tag{3.102}$$

式 (3.102) の分子 $P(adv|ADV[h])$ は，ある副詞 ADV が用言 h を修飾しているとき，その副詞として単語 adv が生成される確率を表わしている．例えば，図 3.19 における 2 つの解釈 (a),(b) においては，以下のような従属係数が考慮される．

$$\text{(a)} \quad D(\text{決っして}\,|ADV[\text{妥協する}]) \tag{3.103}$$

$$\text{(b)} \quad D(\text{決っして}\,|ADV[\text{ない}]) \tag{3.104}$$

副詞「決っして」は形容詞「ない」に係る場合が多いので，これらの従属係数を適切に学習することができれば，$D(\text{決っして}\,|ADV[\text{ない}])$ の値は $D(\text{決っして}\,|ADV[\text{妥協する}])$ よりも十分大きくなり，結果として正しい解釈 (b) に高い生成確率が与えられる．特に呼応の副詞などは係り先の用言との間の語彙的従属関係が強いので，このような従属係数は曖昧性解消に大きく貢献すると期待できる．

以上述べたように，今までに考慮しなかった語彙的従属関係についても，式 (3.93)，式 (3.97)，式 (3.102) のような新たな従属係数を加えることにより，容易に統合的確率言語モデルの中に組み込むことができる．これは，語彙的従属関係を局所化し構文的優先度などの他の統計情報と独立に学習し，また異なる種類の語彙的従属関係は異なる従属係数に独立に学習する，本研究で提案した統合的確率言語モデルの利点である．また，どのような単語を生成する際に単語生成文脈として何を考慮すればよいのかは，3.1.2.2 で述べたように単語生成文脈決定規則によって決定する．したがって，新たな語彙的従属関係を確率モデルに反映させるには，それに対応した新たな単語生成文脈決定規則を追加すればよい．このように，単語生成文脈決定規則の追加・削除により，統合的確率言語モデルに組み込む語彙的従属関係を自由に制御することができる．

第4章　最大エントロピー法の効率化

統計的自然言語処理において，曖昧性解消のための統計情報を学習する際にしばしば問題となるのはデータスパースネス問題である．データスパースネス問題とは，統計情報を学習するためのデータ量が十分にないときに発生する問題である．例えば，ある品詞 l から単語 w の生成確率 $P(w|l)$ を学習する際に，訓練データに一度も現われない単語 x については，その生成確率 $P(x|l)$ は 0 として学習される．したがって，単語の生成確率 $P(w|l)$ を基に曖昧性解消のためのスコアを計算する場合，正解となる解析結果に単語 x が含まれるときには，そのスコアは低く見積られる．このように，訓練データに一度も現われない事象や低頻度でしか現われない事象に対しては，それらの事象に対する統計情報を正しく学習できないことがある．このようなデータスパースネス問題は，統計情報の学習に要するパラメタの数が多ければ多いほど，また統計情報の学習に用いるデータ量が少なければ少ないほど発生しやすい．

このような訓練データに一度も現われない事象や低頻度の事象に対して，その統計情報を推測するのがスムージングと呼ばれる手法である．例えば，先ほどの例においては，訓練データに一度も現われない単語 x に対してある低い出現頻度 γ を与えることにより，生成確率 $P(x|l)$ は 0 ではなく小さい確率値を持つようになる．統計的自然言語処理においては，データスパースネス問題に対処する様々なスムージング手法が提案されている．

このようなスムージング手法の中でも，近年注目されているのが最大エントロピー法 [5] である．最大エントロピー法とは，訓練データから条件付き確率 $P(t|h)$ を推定するアルゴリズムである．1.1 節で述べたように，この最大エントロピー法は自然言語処理に適したスムージング手法であると考えられ，実際に自然言語処理に応用した研究例も報告されている．

- 形態素解析に応用した研究 [72]
- 構文解析に応用した研究 [70, 73]
- PP-attachment 問題に応用した研究 [71]
- 機械翻訳に応用した研究 [5]
- 文境界の認定[1]に応用した研究 [78]
- bi-gram 確率からの n-gram 確率の推定に応用した研究 [20, 21]
- 下位範疇化フレームの自動獲得に応用した研究 [104]
- 対訳単語対の抽出に応用した研究 [81]

最大エントロピー法を自然言語処理に応用する際に問題となるのは，確率モデルの推定に要する計算量が非常に多いということである．このことは，語彙的従属関係を反映する確率モデルなど，推定パラメタ数の多い確率モデルを最大エントロピー法を用いて推定する際の障害となっている．3 章で述べたように，語彙的従属関係は構文解析の曖昧性解消に大きく貢献する統計情報である．ところが，語彙的従属関係は単

[1] 文境界の認定とは，入力としてテキストが与えられたときに，それらの文の境界を決定することを指す．

語の共起関係に関する統計情報であり，また単語の数は非常に多いため，推定パラメタの数も膨大になる．したがって，語彙的従属関係を現在利用可能な言語資源から直接学習することはほとんど不可能に近く，何らかのスムージングを行わなければならない．このとき，語彙的従属関係の学習に，自然言語処理に適した特性を持つと考えられる最大エントロピー法を適用しようとしても，確率モデルの推定に非常に多くの計算量を要するために一般に困難である．これを行うためには，最大エントロピー法による確率モデルの推定に要する計算量を抑制する必要がある．本研究においては，語彙的従属関係のようなパラメタ数の多い統計情報のスムージングに最大エントロピー法を適用することができるように，最大エントロピー法による確率モデル推定アルゴリズムを効率化するいくつかの手法を提案する．

まず，4.1 節では，最大エントロピー法による確率モデル推定の基本原理について概説する．4.2 節では，訓練データから確率モデルを推定するアルゴリズムである Generalized Iterative Scaling アルゴリズム [19] (以下，GIS アルゴリズム) について説明し，これを効率化する 2 つの手法を提案する．4.3 節では，確率モデルの推定に有効な素性を選択する素性選択アルゴリズム [5] について説明し，これを効率化する 4 つの手法を提案する．また 4.4 節では，確率モデルの推定に有効な素性を少ない計算量で選択することを目的に，素性選択アルゴリズムに代わる新たな手法を提案する．最後に 4.5 節では，本研究で提案したアルゴリズムを計算機上に実装し，最大エントロピー法の実際の応用例として助詞の生成確率を推定する．また，既存の素性選択アルゴリズムと 4.4 節で提案する手法の比較実験についても述べる．

4.1 最大エントロピー法

最大エントロピー法とは，事象 t と h が同時に出現する頻度 $O(t,h)$ を訓練データとして，条件付き確率 $P(t|h)$ で表わされる確率モデルを推定するアルゴリズムである．ここで，h は履歴事象 (history) と呼ばれ，条件付き確率の前件となる事象である．一方 t は目標事象 (target) と呼ばれ，確率モデルが予測する事象である．以下，最大エントロピー法の原理について簡単に説明する．

最大エントロピー法では，全ての事象の組 (t,h) について，確率 $P(t|h)$ の値を式 (4.1) によって計算する．

$$P(t|h) = \frac{\prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)}}{\sum_t \prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)}} \tag{4.1}$$

ここで $f_i(t,h)$ は素性 (feature) と呼ばれ，目標事象，履歴事象の組に対して 1 または 0 を返す任意の関数であり，また F はこのような素性の集合である．一方，α_i は素性パラメタと呼ばれるものである．

次に，素性の期待値 $E(f_i)$, $\hat{E}(f_i)$ を式 (4.2), (4.3) のように定義する．

$$E(f_i) = \sum_{t,h} P(t,h) f_i(t,h) \simeq \sum_{t,h} \hat{P}(h) P(t|h) f_i(t,h) \tag{4.2}$$

$$\hat{E}(f_i) = \sum_{t,h} \hat{P}(t,h) f_i(t,h) \tag{4.3}$$

但し，

$$\hat{P}(t,h) = \frac{O(t,h)}{\sum_{t,h} O(t,h)} \tag{4.4}$$

$$\hat{P}(h) = \sum_t \hat{P}(t,h) \tag{4.5}$$

$E(f_i)$ は，最大エントロピー法によって推定された確率モデル $P(t|h)$ において，素性 f_i が事象 (t,h) に対して 1 を返す期待値を表わしている．一方 $\hat{E}(f_i)$ は，訓練データによって最尤推定された確率モデル $\hat{P}(t,h)$

において，素性 f_i が事象 (t,h) について 1 を返す期待値を表わしている．言い換えれば，$E(f_i)$ とは素性 f_i が 1 を返す事象の推定された確率モデルにおける同時確率 $P(t,h)$[2]の総和であり，$\hat{E}(f_i)$ とは素性 f_i が 1 を返す事象の訓練データにおける同時確率 $\hat{P}(t,h)$ の総和である．

確率モデルの推定は，

1. 素性に関する式 (4.6) の制約を満たしつつ，

$$\forall f_i \in F \qquad E(f_i) = \hat{E}(f_i) \tag{4.6}$$

2. $P(t|h)$ のエントロピー $H(P)$ が最大となるように，

$$H(P) = -\sum_h \hat{P}(h) \sum_t P(t|h) \log P(t|h) \tag{4.7}$$

素性パラメタ α_i を反復推定することにより行われる．式 (4.6) で表わされる素性に関する制約とは，訓練データにおける素性の期待値と確率モデルにおける素性の期待値が等しいという制約である．一方，式 (4.7) に示す確率モデルのエントロピー $H(P)$ が最大になるということは，推定される条件付き確率 $P(t|h)$ が一様分布に近くなることを意味する．すなわち，最大エントロピー法による確率モデルの推定とは，ある素性が 1 を返す事象集合についてはその各事象の同時確率 $P(t,h)$ の和を訓練データのそれに近付け，かつ確率分布が一様分布になるべく近くなるように，各素性のパラメタ α_i を推定することである．このような素性パラメタ α_i の反復推定を行うアルゴリズムのひとつが GIS アルゴリズムである．

このことを具体例を用いて説明する．最大エントロピー法によって推定する確率モデルとして，ある動詞 v のヲ格の格要素として名詞 n が現われる確率 $P(n|N[s(v,\text{を})])$ を考える．簡単のため，推定する確率モデルの目標事象を名詞 n，履歴事象を動詞 v とし，推定する確率モデルを以下のように記述する．

$$P(n|N[s(v,\text{を})]) \;=\; P(n|v) \tag{4.8}$$

この確率モデルの推定に用いる素性の例を式 (4.9) に挙げる．

$$ftr(n,v) = \begin{cases} 1 & if \;\; n \in C_{\text{食物}} \;\; and \;\; v = \text{食べる} \\ 0 & otherwise \end{cases} \tag{4.9}$$

ここで，$C_{\text{食物}}$ は “食物” を表わす名詞の意味クラスであり，“$n \in C_{\text{食物}}$” は名詞 n が意味クラス $C_{\text{食物}}$ に属することを意味する．このような素性が F の要素として与えられたとき，履歴事象 v が「食べる」であるときの確率モデルは図 4.1 のように推定される．

図 4.1 における実線は推定された確率モデルにおける確率分布を表わし，点線は訓練データにおける確率分布を表わしている．もし，訓練データにおいて「食べる」という動詞のヲ格として “食物” を表わす名詞が頻繁に出現していれば，“食物” を表わす名詞 $n \in C_{\text{食物}}$ の出現確率が高くなるように確率モデルも推定される．これに加えて，確率モデルはエントロピーが最大に，すなわち一様分布になるべく近くなるように推定される．このため，“食物” を表わす名詞の出現確率，およびそれ以外の名詞の出現確率は全て等しくなるように推定される．

式 (4.9) の素性は「食べる」という特定の履歴事象の確率分布を決めるものであったが，複数の履歴事象の確率分布を決める式 (4.10) のような素性も記述できる．

$$ftr(n,v) = \begin{cases} 1 & if \;\; n \in C_{\text{食物}} \;\; and \;\; v \in C_{\text{食べる}} \\ 0 & otherwise \end{cases} \tag{4.10}$$

[2] これは $\hat{P}(h)P(t|h)$ によって近似されている．

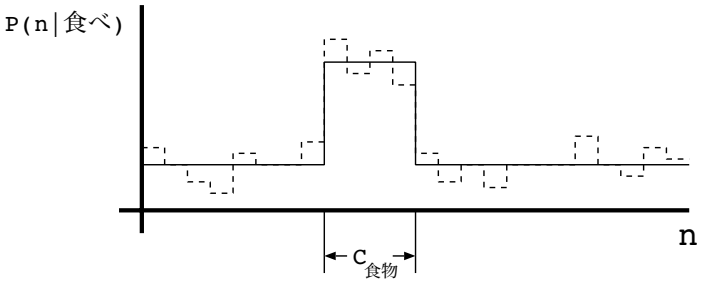


図 4.1: 動詞「食べる」における確率分布

この素性は，“食べる” という行為を表わす動詞の意味クラス $C_{食べる}$ に属する動詞 v が与えられたときに，そのヲ格に “食物” を表わす名詞の意味クラス $C_{食物}$ に属する名詞が出現するという事象 (図 4.2 の斜線部) の確率和が訓練データにおける確率和と等しくなるように確率モデルを推定する働きをする．

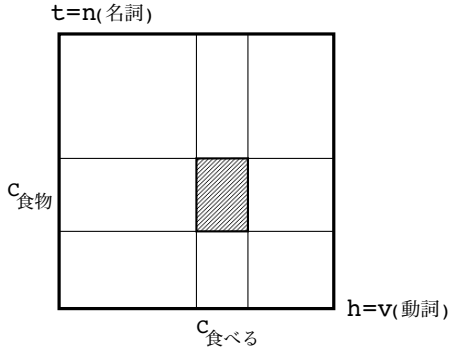


図 4.2: 素性 (4.10) が 1 を返す事象の範囲

したがって，訓練データにおいて $C_{食べる}$ に属する動詞のヲ格に $C_{食物}$ に属する名詞が多く出現しているなら，図 4.2 の斜線部に相当する事象の確率も高くなるように確率モデルが推定される．

以上述べたように，素性は事象の組 (t, h) に対して 1 または 0 を返す関数であり，訓練事象全体における部分集合を限定し，その部分集合に属する事象の推定された確率モデルにおける確率和が訓練データにおける確率和と一致しなければならないという制約として働く．このとき，各素性が限定する事象の部分集合は，図 4.2 に示すような長方形でなくてはならない．すなわち，素性が事象の組 (t, h) に対して 1 または 0 を返す条件を記述する際には，目標事象に対する条件と履歴事象に対する条件を独立に記述しなければならない．但し，素性 (4.10) のように，両者の論理積を条件としても構わない．

最大エントロピー法は，以下のような優れた特長を持つ．

- 目標事象，履歴事象の両方を抽象化した素性を取り扱うことができる

  目標事象，履歴事象のどちらか片方を抽象化した素性 (例えば素性 (4.9)) しか用いることができない場合には，確率モデルの推定パラメタの数が多くなり効率が悪い．例えば，式 (4.10) のような素性を，

目標事象のみしか抽象化しない素性を用いて表わすと以下のようになる．

$$\forall\ v' \in C_{\text{食べる}} \qquad ftr(n,v) = \begin{cases} 1 & if\ \ n \in C_{\text{食物}}\ \ and\ \ v = v' \\ 0 & otherwise \end{cases} \tag{4.11}$$

すなわち，$C_{\text{食べる}}$ に属する動詞 v' の数だけ素性を用意しなければならないが，これは推定する素性パラメタ α_i の数が多くなるために効率が悪い．ところが，最大エントロピー法においては，素性 (4.10) のような目標事象・履歴事象の両方を抽象化する素性を用いることができるので，推定する素性パラメタの数も 1 つでよい．

- 素性が 1 を返す事象集合の大きさに制限がない

  これは，図 4.2 において，素性が 1 を返す事象の集合として任意の大きさの長方形を指定できるということに相当する．例えば，$C_{\text{食物}}$ よりも抽象度の大きい $C_{\text{物体}}$ のような意味クラスを用いて，式 (4.12) のような素性を F の要素としても構わない．

$$ftr(n,v) = \begin{cases} 1 & if\ \ n \in C_{\text{物体}}\ \ and\ \ v = \text{作る} \\ 0 & otherwise \end{cases} \tag{4.12}$$

  ここで重要なのは，様々な大きさの事象集合に対して 1 を返す素性が混在しても構わないという点である．これにより,「食べる」という動詞については抽象化レベルの低い $C_{\text{食物}}$ という意味クラスを用いた素性 (4.9) を使い,「作る」という動詞については抽象化レベルの高い $C_{\text{物体}}$ という意味クラスを用いた素性 (4.12) を使うこともできる．このように，素性が 1 を返す事象集合の大きさを任意に変えることにより，より柔軟なスムージングが可能となる．

- 素性が 1 を返す事象の集合に重なりがあっても良い

  例えば，図 4.3 において，素性 f_a と f_b が 1 を返す事象集合に重なりがあっても，最大エントロピー法はそれぞれの素性が式 (4.6) の制約を満たすように確率モデルを推定することができる．

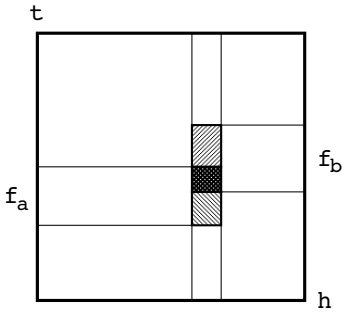


図 4.3: 1 を返す事象集合に重なりがある素性

このことは，自然言語処理への応用に適した特性のひとつであると考えられる．なぜなら，素性 (4.9) や素性 (4.10) のように意味クラスを用いて素性を定義する場合，様々な視点による意味クラスを同時に利用することができるからである．例えば,「学校」という単語は，建造物という視点から見れば $C_{\text{建物}}$ という意味クラスに属し，組織という視点から見れば $C_{\text{組織}}$ という意味クラスに属している．このような様々な視点から見た意味クラスを同時に利用する場合には，それらの素性が 1 を返す事象集

合に重なり (例えば，単語「学校」は両方の意味クラスに属する) ができるが，最大エントロピー法はこのような場合でも確率モデルを推定することが可能である．

素性が 1 を返す事象集合に重なりがない場合には，例えば式 (4.13) のような簡単な式を用いて確率モデルを推定することによって，非常に少ない計算量でスムージングを行うことができる．

$$P(t,h) \;=\; \frac{\sum_{\forall t_i,h_j \;\; f(t_i,h_j)=1} \hat{P}(t_i,h_j)}{N} \tag{4.13}$$

但し，N は $f(t_i,h_j)=1$ となる事象の組 (t_i,h_j) の数

式 (4.13) は，素性 f が 1 を返す事象集合 (t_i,h_j) の訓練データにおける確率 $\hat{P}(t_i,h_j)$ の平均を $P(t,h)$ とすることを意味する．しかしながら，このようにして推定できる確率モデルとしては，素性が 1 を返す事象集合に重なりがないという条件を満たさなければならないために比較的単純なものに限定される．これに対して，最大エントロピー法はより一般的な確率モデル，すなわち素性が 1 を返す事象集合に重なりがあるような確率モデルを推定することができる．

このように，最大エントロピー法は優れた特長を持っているが，以下に述べるような問題も存在する．まず第一に，確率モデルの推定に用いる GIS アルゴリズムの計算量が非常に多いということである．これは最大エントロピー法を実際に自然言語処理に応用する際の障害となる．もうひとつは，確率モデルの推定に用いる素性の集合 F をどのように決定すればよいのかということである．本節で述べた最大エントロピー法の原理から容易に推察できるように，推定される確率モデルの性質は素性集合 F に大きく依存する．この F を決定する方法としては，まず素性候補の集合 S を作り，その中の 1 つの素性を採用したときの確率モデル全体のエントロピーの変化量を調べ，その変化量の最も大きい素性を確率モデルの推定に有効であると判断して F に加え，これを繰り返す素性選択アルゴリズムと呼ばれる方法が提案されている [5]．しかしながら，この方法は，1 つの素性を F に加える度に，全ての素性候補についてそれを素性として採用したときの確率モデル全体のエントロピーの変化量を計算し，また GIS アルゴリズムによって確率モデルの推定をやり直すため，素性選択に要する計算量が非常に多いという問題点がある．

4.2 GIS アルゴリズムの効率化

本節では，訓練共起データ $O(t,h)$ と素性集合 F をもとに，各素性のパラメタ α_i を反復推定する GIS アルゴリズムの概要と，それを効率化する手法について述べる．

まず，GIS アルゴリズムの概要を以下に示す．

【GIS アルゴリズム】

1. 訓練データにおける素性の期待値 $\hat{E}(f_i)$ を計算する．
2. 全ての素性パラメタの初期値 $\alpha_i^{(0)}$ を 1 とする．
3. 式 (4.1) より，与えられたパラメタ $\alpha_i^{(n)}$ における $P^{(n)}(t|h)$ を計算する．
4. 式 (4.2) より，確率モデルにおける素性の期待値 $E^{(n)}(f_i)$ を計算する．
5. α_i を以下のように更新する．

$$\alpha_i^{(n+1)} = \alpha_i^{(n)} \left[\frac{\hat{E}(f_i)}{E^{(n)}(f_i)} \right]^{\frac{1}{C}} \tag{4.14}$$

6. α_i が収束するまで 3.〜5. を繰り返す

式 (4.14) における C は補完定数と呼ばれる定数である．GIS アルゴリズムは，収束条件として，あらゆる目標事象・履歴事象の組 (t,h) に対して，各素性 f_i の返す値の和，すなわち (t,h) に対して 1 を返す素性の数は等しくなければならないという制約を持つ．

$$\forall t, \forall h \qquad \sum_{i=1}^{|F|} f_i(t,h) = C \tag{4.15}$$

ところが，通常は式 (4.15) の制約は満たされていないため，以下のような補完定数 C と補完素性 f_c を導入する [74].

$$C = \max_{t,h} \sum_{i=1}^{|F|} f_i(t,h) \tag{4.16}$$

$$f_c(t,h) = C - \sum_{i=1}^{|F|} f_i(t,h) \tag{4.17}$$

$$\tag{4.18}$$

補完素性 f_c を素性集合 F に加えることにより式 (4.15) の制約は満たされ，GIS アルゴリズムも必ず収束する．

GIS アルゴリズムの 1 回の反復に要する計算量は $O(\,|T|\,|H|\,|F|\,)$ である．但し，T は目標事象の集合，H は履歴事象の集合，F は素性の集合である．ところが，推定パラメタ数の多い確率モデルを学習しようとする場合には，この計算量は無視できないほど大きくなる．例えば，4.1 節において，動詞 v が与えられたときに，そのヲ格に表われる名詞 n の出現する確率を与える確率モデル $P(n|v)$ を例として挙げた．この確率モデルを GIS アルゴリズムを用いて推定する場合，日本語における動詞の数は約 10,000($=|H|$)，名詞の数は約 200,000($=|T|$) 程度であるので，1 回の反復推定に要する計算量だけでもかなり大きい．

次に，GIS アルゴリズムを効率化する 2 つの手法を以下に示す．

- ある履歴事象 h について 1 を返す素性の集合を F_h とする．履歴事象 h_1 と h_2 において，F_{h_1} と F_{h_2} が等しいなら，式 (4.1) より確率分布 $P(t|h_1)$ と $P(t|h_2)$ は全く同一となる．

  $$\forall t \in T \qquad P(t|h_1) = P(t|h_2) \tag{4.19}$$

  したがって,【GIS アルゴリズム】の Step 3. の $P(t|h)$ の計算において，F_h が等しい履歴事象については，その中の 1 つの履歴事象 h_1 について $P(t|h_1)$ を計算すれば，その他の履歴事象については計算を省略できる．

- ある事象 (t_1,h) と (t_2,h) において，これらに対して 1 を返す素性の集合が全く同じならば，式 (4.20) に示すように，式 (4.1) の分子の項は全く同じ値になるはずである．

  $$\prod_{i=1}^{|F|} \alpha_i^{f_i(t_1,h)} = \prod_{i=1}^{|F|} \alpha_i^{f_i(t_2,h)} \tag{4.20}$$

  したがって,【GIS アルゴリズム】の Step 3. の $P(t|h)$ の計算において，先ほどと同様に同じ値となる項の計算を省略することができる．

以上の手法により，計算量がどの程度削減されるのかについては素性集合 F に依存する．一般に，素性集合 F の要素数が少なければ少ないほど，計算量も大幅に削減される．

4.3 素性選択アルゴリズムの効率化

4.1 節で述べたように，最大エントロピー法によって推定される確率モデルの品質は素性集合 F に大きく依存する．したがって，素性集合 F をどのように決定するかは，最大エントロピー法における最も重要な問題である．この F を決定する方法としては，素性選択アルゴリズムと呼ばれるものが提案され [5]，また実際にこれを応用した研究もいくつか報告されている [5, 71, 81]．素性選択アルゴリズムとは，与えられた素性の候補の集合 S から，確率モデルに採用するべき素性集合 F を選択するアルゴリズムである．これは，S の中から確率モデルのログ尤度 (log likelihood) を最も増大させる素性を 1 つ選択し，それを F に追加するといった操作を繰り返すことによって行う．ログ尤度とは，式 (4.21) で定義され，エントロピーと同じく確率モデルが一様分布にどれだけ近いか (ログ尤度が低くなればなるほど確率モデルは一様分布に近付く) を表わす指標である．

$$L(P) = \sum_{t,h} \hat{P}(t,h) \log P(t|h) \tag{4.21}$$

素性選択アルゴリズムの詳細を以下にまとめる．

【素性選択アルゴリズム】

1. F を空集合とする．
2. 素性候補の集合 S の各要素 f_j について，f_j を F に加えたときの f_j のパラメタ α_j を求める．これは GIS アルゴリズムにより反復推定する．
3. f_j を F に加えたときのログ尤度の増分 $\Delta L(f_j)$ を計算する．$\Delta L(f_j)$ の最も大きい素性を 1 個選択し，それを確率モデルの素性として新たに F に追加する．
4. 新しい F における各素性のパラメタ α_i を GIS アルゴリズムを用いて再推定する．
5. 素性を加えたときの確率モデルのエントロピーの変化量がある閾値 T_{ent} 以下になるまで 2.〜4. を繰り返す．

この素性選択アルゴリズムは最小記述長原理 [49] (Minimum Description Length Priciple，以下 MDL 原理) による素性選択とほぼ一致する．MDL 原理においては，確率モデルの良さを表わす評価尺度としてモデル記述長とデータ記述長を用いる．モデル記述長は推定された確率モデルの複雑さを表わし，データ記述長は推定された確率モデルと訓練データにおける確率モデルとの近さを表わす．そして，このモデル記述長とデータ記述長の和が最小となるように，すなわちなるべく単純な確率モデルでかつ訓練データにおける確率モデルに近い確率分布を持つように，確率モデルに与える素性を決定する．素性選択アルゴリズムにおいて，素性を 1 個増やすことは確率モデルを複雑にし，MDL 原理におけるモデル記述長を増やすことに対応する．また，そのときに確率モデルのログ尤度が大きくなることは，MDL 原理におけるデータ記述長を減らすことに対応する．したがって，素性を 1 個増やしたときのログ尤度の増分が最大となる素性を選択することは，MDL 原理におけるモデル記述長とデータ記述長の和が最小となる確率モデルを選択することに等しい．

MDL 原理を自然言語処理に応用した例としては，動詞の格フレームを学習する研究 [49, 53, 54] や単語のクラスタリングを行った研究 [52] などがある．これらの研究においては，素性が 1 を返す事象集合に重なりがないことを前提にしている．これに対し，最大エントロピー法における素性選択アルゴリズムは，素性が 1 を返す事象の集合に重なりがあるような一般的な確率モデルにも適用することが可能である[3]．

この素性選択アルゴリズムは，全ての素性候補について，それを素性として採用したときのログ尤度を求めるために素性パラメタを GIS アルゴリズムによって反復推定し，またログ尤度の増分が最も大きい素性

[3] MLD 原理そのものがこのような確率モデルの推定に適用できないというわけではないが，実際に自然言語処理における研究例として報告されたことはない．

を素性集合 F に加える度に GIS アルゴリズムによって確率モデル全体の推定をやり直すため，計算量が非常に多いという問題がある．そこで，素性選択アルゴリズムを効率化する 4 つの手法を提案する．

- 【素性選択アルゴリズム】の Step 3. において，素性候補の集合 S の中で $\Delta L(f_j)$ の値が最も大きい素性を f_1, 2 番目に大きい素性を f_2 とする．このとき，まず f_1 のみを F に加えて確率モデルの推定をやり直してから (Step 4.)，再び素性候補の集合 S 中の各素性のログ尤度の増分 (これを $\Delta L'(f_j)$ とする) を計算する．

  ここで，f_1, f_2 が条件 (4.22) を満たす場合を考える．

$$T_{f_1} \cap T_{f_2} = \emptyset \ \text{ and } \ H_{f_1} \cap H_{f_2} = \emptyset \tag{4.22}$$

  式 (4.22) において，T_f は素性 f が 1 を返す目標事象の集合であり，H_f は素性 f が 1 を返す履歴事象の集合である．本研究においては，式 (4.22) の条件を満たす 2 つの素性 f_1 と f_2 は互いに独立であると定義する．f_1 と f_2 が互いに独立であることは，図 4.4 に示すように，2 つの素性が 1 を返す事象集合に重なりがないことを意味する．

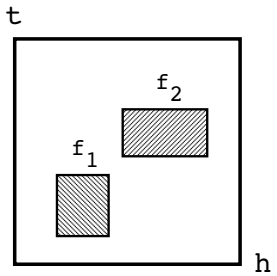


図 4.4: 互いに独立な素性

  f_1 と f_2 が互いに独立な場合，すなわち全く異なる事象に対する確率分布を決定する素性である場合は，まず F に f_1 を追加してから確率モデルを推定し，f_2 に対するログ尤度の増分 $\Delta L'(f_2)$ を計算しても，その値は $\Delta L(f_2)$ とほぼ変わらない，すなわち $\Delta L(f_2) \simeq \Delta L'(f_2)$ であると予想される．したがって，f_1 の次に追加される素性として f_2 が選ばれる可能性が高い．そこで，F に素性を 1 個ずつ追加する代わりに，ΔL の値が大きく，かつ互いに独立である上位 N_f 個の素性を F に同時に追加するように【素性選択アルゴリズム】を修正した．これにより，素性選択に要する時間を約 $1/N_f$ に短縮できる．

- 【素性選択アルゴリズム】の Step 2. においては，素性候補の集合 S に含まれる全ての要素 f_i について，$\{\alpha_i\}$, α_j の全てのパラメタを GIS アルゴリズムを用いて再推定している．ここで，$\{\alpha_i\}$ は既に選択された素性集合 F 中の各素性に対応したパラメタであり，α_j はこれから素性として選択するべき素性候補の集合 S のひとつの要素 f_j に対応したパラメタを表わす．ところが，これは GIS アルゴリズムによる反復推定に要する計算量が多いために効率が悪い．そこで，素性パラメタ $\{\alpha_i\}$ の値は変化させず，α_j のみを再推定するように修正した GIS アルゴリズムを適用することによって α_j の値を近似推定する．この修正された GIS アルゴリズムの詳細を以下に示す．

  1. $\alpha_j^{(0)} = 1$ とする．

2. $P^{(n)}(t|h)$ を計算する．

$$P^{(n)}(t|h) = \frac{\prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \times \alpha_j^{(n)f_j(t,h)}}{\sum_t \prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \times \alpha_j^{(n)f_j(t,h)}} \tag{4.23}$$

3. $E^{(n)}(f_j)$ を計算する．
4. α_j の値を更新する．

$$\alpha_j^{(n+1)} = \alpha_j^{(n)} \left[\frac{\hat{E}(f_j)}{E^{(n)}(f_j)} \right]^{\frac{1}{C}} \tag{4.24}$$

5. α_j の値が収束するまで繰り返す．

素性集合 F の要素として既に選択された素性に対応したパラメタ $\{\alpha_i\}$ については，式 (4.24) に示した反復推定に要する計算を省略することにより，パラメタ推定に要する計算量を大幅に削減することができる．

- α_j のみ推定するように修正した GIS アルゴリズムにおいては，式 (4.23) に示した $P^{(n)}(t|h)$ の計算は以下のようになる．

$$\begin{aligned} P^{(n)}(t|h) &= \frac{\prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \times \alpha_j^{(n)f_j(t,h)}}{\sum_t \prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \times \alpha_j^{(n)f_j(t,h)}} \\ &= \frac{\prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \times \alpha_j^{(n)f_j(t,h)}}{D(\overline{T}_{f_j}) \ + \ D(T_{f_j}) \times \alpha_j^{(n)}} \end{aligned} \tag{4.25}$$

$$D(T_{f_j}) = \sum_{t \in T_{f_j}} \prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \tag{4.26}$$

$$D(\overline{T}_{f_j}) = \sum_{t \in \overline{T}_{f_j}} \prod_{i=1}^{|F|} \alpha_i^{f_i(t,h)} \tag{4.27}$$

ここで，T_{f_j} は f_j が 1 を返す目標事象の集合であり，$\overline{T}_{f_j}$ は f_j が 1 を返さない目標事象の集合である．すなわち，$T_{f_j} \cup \overline{T}_{f_j} = T$ である．$D(T_{f_j})$, $D(\overline{T}_{f_j})$ の値は α_j の値に全く依存しないので，これらは α_j の推定の第 1 回目の反復時にのみ計算すれば，第 2 回目以降の反復時には計算を省略できる．

- S 中の 2 つの素性の候補 f_1 と f_2 について，それらが 1 を返す目標事象の集合が等しい場合 ($T_{f_1} = T_{f_2}$) を考える．このとき，f_1, f_2 のパラメタ推定において，式 (4.25) の項 $D(T_{f_j})$, $D(\overline{T}_{f_j})$ の値は全く等しい．したがって，ΔL を計算する素性候補の順序を T_{f_j} によってソートすることにより，一部の素性の候補については項 $D(T_{f_j})$, $D(\overline{T}_{f_j})$ の計算を完全に省略することができる．

4.4 素性効用に基づく素性選択

4.3 節で述べた素性選択アルゴリズムは，その計算量を削減するいくつかの手法を提案したものの依然として計算量が多く，パラメタ数の多い確率モデルの推定に適用するのは難しい．これは，素性選択アルゴリズムが，素性候補の集合 S から素性を選択して素性集合 F に追加する度に，GIS アルゴリズムによって確率モデルの推定をやり直すことに起因する．

本節では，素性選択アルゴリズムと同様に，素性候補の集合 S から確率モデルの推定に有効な素性の集合 F を選択する新しい手法を提案する．本手法では，まず素性の候補 f が確率モデルの推定にどれだけ有効であるかを表す尺度として，素性効用 $U(f)$ を定義する．そして，素性候補の集合 S が与えられたときに，それぞれの素性候補に対して $U(f)$ を計算し，この値の大きい N 個の素性を素性集合 F の要素として選択する．

素性効用 $U(f)$ の定義を以下に示す．

$$U(f) = \max\,(\,U_T(f)\,,\,U_H(f)\,) \tag{4.28}$$

$$U_T(f) = H(P_{\{f^T\}}) - H(P_{\{f,f^T\}}) \tag{4.29}$$

$$U_H(f) = H(P_{\{f^H\}}) - H(P_{\{f,f^H\}}) \tag{4.30}$$

ここで，$P_{\{f,\cdots\}}$ は素性集合 F が $\{f,\cdots\}$ であるときの確率モデルを表わす．また，f^T および f^H は，素性 f をもとに以下のように定義される素性である．

$$f^T(t,h) = \begin{cases} 1 & if \quad \exists h,\ f(t,h)=1 \\ 0 & otherwise \end{cases} \tag{4.31}$$

$$f^H(t,h) = \begin{cases} 1 & if \quad \exists t,\ f(t,h)=1 \\ 0 & otherwise \end{cases} \tag{4.32}$$

f^T は，履歴事象 h に対して素性 f が 1 を返すことがあれば，目標事象 t がどのような事象であっても 1 を返す素性である．一方 f^H は，目標事象 t に対して素性 f が 1 を返すことがあれば，履歴事象 h がどのような事象であっても 1 を返す素性である．f と f^T および f^H との関係を図 4.5 に示す．すなわち，f^T および f^H は，f が 1 を返す事象の範囲を全ての目標事象および履歴事象に拡張した素性である．

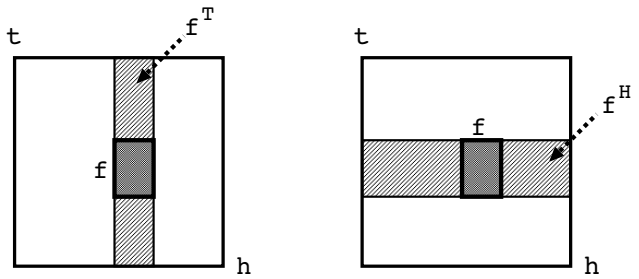


図 4.5: f^T，f^H の定義

次に，式 (4.29) によって定義される $U_T(f)$ の持つ意味について説明する．$U_T(f)$ は，素性集合 F に含まれている素性が f^T のみであるときに，さらに素性 f を F に加えたときのエントロピーの変化量を表わしている．4.1 節で述べたように，素性 f^T は図 4.5 における斜線部の事象の確率の和を訓練データにおける確率の和に等しくし，かつ斜線部内の各事象の確率を全て等しくする働きを持つ．したがって，素性集合 F の要素として f^T のみが選択されているとき，斜線部内の事象の確率 $P(t,h)$ は全て等しくなるように学習されている．このとき，さらに f を F に加えた場合を考えてみよう．素性 f は，f が 1 を返す事象集合 (図 4.5 における灰色の部分) の事象の確率の和を訓練データにおける確率和に等しくし，かつ各事象の確率を全て等しくする働きを持つ．しかしながら，もし訓練データにおける斜線部内の事象の確率がほぼ等しい場合には，f が 1 を返す事象集合における確率分布は素性 f を加える前から訓練データの確率分布に近くなっているため，F に f を加えても確率分布はそれほど変化せず，確率モデル全体のエントロピーの

変化量も小さい．したがって，このような素性を F に加えてもその効果は少なく，学習に有効であるとは言えない．これに対し，図 4.5 において，灰色の部分の事象の訓練データにおける確率分布が周りの斜線部の部分と著しく異なる場合には，F に f を加えることにより，素性 f が 1 を返す事象集合の確率も訓練データに近づくように大きく変化するため，確率モデル全体のエントロピーの変化量も大きい．このように，$U_T(f)$ は，素性 f が 1 を返す事象集合の確率分布がその周辺と比べてどの程度違いがあるかを示している．式 (4.30) で定義される $U_H(f)$ についても同様のことが言える．

本研究においては，周囲と大きく異なる確率分布を持つ事象の範囲を限定する素性は学習に有効であるとみなし，$U_T(f)$ と $U_H(f)$ の最大値である効用 $U(f)$ (式 (4.28)) を，素性 f が確率モデルの学習にどの程度有効であるかの指標とする．また，$U_T(f)$ または $U_H(f)$ を計算する際には，GIS アルゴリズムや素性選択アルゴリズムのような反復推定を行う必要はなく，訓練データとなる出現頻度 $O(t,h)$ から直接計算することできる．したがって，素性効用に基づいた素性選択に要する計算量は，素性選択アルゴリズムのそれに比べて非常に小さい．$U_T(f)$ および $U_H(f)$ の具体的な計算方法については 4.4.1 項および 4.4.2 項で説明する．

4.4.1 $U_T(f)$ の計算方法

$U_T(f)$ は，式 (4.29) に示した通り，素性集合 F の要素として f^T のみを考えたとき (図 4.6 (a)) の確率モデル全体のエントロピー $H(P_{\{f^T\}})$ と，素性集合 F の要素として f と f^T の 2 つを考えたとき (図 4.6 (b)) の確率モデル全体のエントロピー $H(P_{\{f,f^T\}})$ との差である．

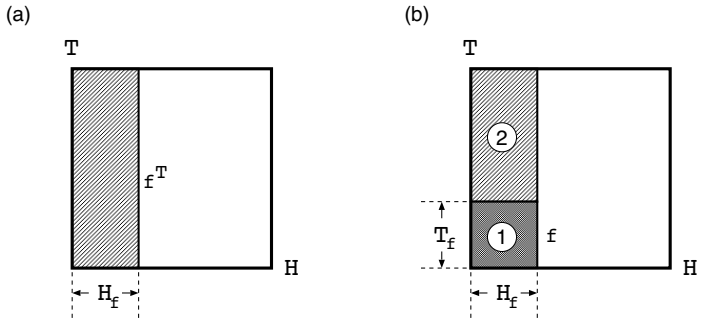


図 4.6: $U_f(T)$ の計算

図 4.6 において，T および H はそれぞれ目標事象および履歴事象の全体集合を，また T_f および H_f はそれぞれ素性 f が 1 を返す目標事象および履歴事象の部分集合を表わしている．

まず，エントロピー $H(P_{\{f^T\}})$ の計算方法を説明する．

$$H(P_{\{f^T\}}) = -\sum_{h\in H}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) \tag{4.33}$$

$$= -\sum_{h\in H_f}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) - \sum_{h\in \overline{H_f}}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) \tag{4.34}$$

ここで，式 (4.34) 内の項 $P(t|h)$ の値は全て $1/|T|$ となる．なぜなら，最大エントロピー法においては，式 (4.6) の制約を満たす範囲でエントロピーを最大に，すなわち確率分布が一様分布となるように確率モデルが推定されるからである．したがって，全ての目標事象 t について，$P(t|h)$ の値は等しく $1/|T|$ となる．このことから，$H(P_{\{f^T\}})$ の計算を以下のように簡略できる．

$$
\begin{aligned}
H(P_{\{f^T\}}) &= -\sum_{h\in H_f}\hat{P}(h)\sum_{t\in T}\frac{1}{|T|}\log\frac{1}{|T|} - \sum_{h\in\overline{H_f}}\hat{P}(h)\sum_{t\in T}\frac{1}{|T|}\log\frac{1}{|T|} && (4.35)\\
&= -\sum_{h\in H_f}\hat{P}(h)\log\frac{1}{|T|} - \sum_{h\in\overline{H_f}}\hat{P}(h)\log\frac{1}{|T|} && (4.36)\\
&= -\hat{P}(H_f)\log\frac{1}{|T|} - \hat{P}(\overline{H_f})\log\frac{1}{|T|} && (4.37)
\end{aligned}
$$

式 (4.37) において，$\hat{P}(H_f)$ および $\hat{P}(\overline{H_f})$ は，以下に示すように，H_f および $\overline{H_f}$ に属する履歴事象の訓練データにおける出現確率である．

$$
\begin{aligned}
\hat{P}(H_f) &= \sum_{h\in H_f}\hat{P}(h) && (4.38)\\
\hat{P}(\overline{H_f}) &= \sum_{h\in\overline{H_f}}\hat{P}(h) && (4.39)
\end{aligned}
$$

次に，エントロピー $H(P_{\{f,f^T\}})$ の計算方法を説明する．

$$
\begin{aligned}
H(P_{\{f,f^T\}}) &= -\sum_{h\in H}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) && (4.40)\\
&= -\sum_{h\in H_f}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) - \sum_{h\in\overline{H_f}}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) && (4.41)\\
&= -\sum_{h\in H_f}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) - \hat{P}(\overline{H_f})\log\frac{1}{|T|} && (4.42)\\
&= \sum_{h\in H_f}\hat{P}(h)H_{(T|h)} - \hat{P}(\overline{H_f})\log\frac{1}{|T|} && (4.43)
\end{aligned}
$$

式 (4.42) の第 2 項は式 (4.37) の第 2 項と同様に導かれる．また，式 (4.43) の第 1 項における $H_{(T|h)}$ の定義と計算式を以下に示す．

$$
\begin{aligned}
H_{(T|h)} &\stackrel{def}{=} -\sum_{t\in T}P(t|h)\log P(t|h) && (4.44)\\
&= P(T_f|h)H_{(T_f|h)} + P(\overline{T_f}|h)H_{(\overline{T_f}|h)} + H'_{(T_f,\overline{T_f})} && (4.45)
\end{aligned}
$$

式 (4.45) の導出は付録 A に示す．式 (4.43) における $H_{(T|h)}$ は，図 4.7 に示すように，H_f に属するある特定の履歴事象 h についての条件付き確率 $P(t|h)$ のエントロピーを表わしている．以下，式 (4.45) の各項の意味，およびその計算方法について説明する．

- $P(T_f|h)$ および $P(\overline{T_f}|h)$

  次式に示すように，履歴事象が h であるという条件の下で T_f および $\overline{T_f}$ に属する目標事象が出現する確率である．

$$
\begin{aligned}
P(T_f|h) &= \sum_{t\in T_f}P(t|h) && (4.46)\\
P(\overline{T_f}|h) &= \sum_{t\in\overline{T_f}}P(t|h) && (4.47)
\end{aligned}
$$

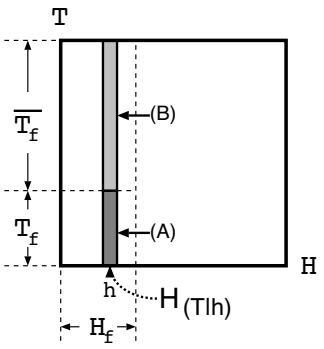


図 4.7: 目標事象を二分したときのエントロピー $H_{(T|h)}$

これらの確率の比は，訓練データにおける確率分布 $\hat{P}(t,h)$ から以下のように計算することができる．

$$P(T_f|h) \ : \ P(\overline{T_f}|h) \ = \ \hat{P}(T_f, H_f) \ : \ \hat{P}(\overline{T_f}, H_f) \tag{4.48}$$

但し，

$$\hat{P}(T_f, H_f) \ = \ \sum_{t \in T_f, h \in H_f} \hat{P}(t,h) \tag{4.49}$$

$$\hat{P}(\overline{T_f}, H_f) \ = \ \sum_{t \in \overline{T_f}, h \in H_f} \hat{P}(t,h) \tag{4.50}$$

なぜなら，最大エントロピー法の原理から，図 4.6 の ① および ② で示した事象集合の推定された確率モデルにおける確率和は訓練データにおける確率和と等しく，かつそれぞれの事象集合に属する個々の事象の確率 $P(t,h)$ は全て等しいため，推定された確率モデルにおける図 4.7 中の (A) の事象集合の確率和と (B) の事象集合の確率和の比 (式 (4.48) の左辺) は，訓練データから最尤推定される確率モデルにおける図 4.6 中の ① の事象集合の確率和と ② の事象集合の確率和の比 (式 (4.48) の右辺) に等しいからである．また，式 (4.48) と $P(T_f|h) + P(\overline{T_f}|h) = 1$ であることから，個々の確率 $P(T_f|h)$, $P(\overline{T_f}|h)$ は以下の式で計算できる．

$$P(T_f|h) \ = \ \frac{\hat{P}(T_f, H_f)}{\hat{P}(T_f, H_f) + \hat{P}(\overline{T_f}, H_f)} \tag{4.51}$$

$$P(\overline{T_f}|h) \ = \ \frac{\hat{P}(\overline{T_f}, H_f)}{\hat{P}(T_f, H_f) + \hat{P}(\overline{T_f}, H_f)} \tag{4.52}$$

- $H_{(T_f|h)}$ および $H_{(\overline{T_f}|h)}$
  履歴事象が h でありかつ出現する目標事象が T_f および $\overline{T_f}$ に属するという条件の下での確率分布 $P(t|T_f,h)$ および $P(t|\overline{T_f},h)$ のエントロピーである．

$$H_{(T_f|h)} \ = \ -\sum_{t \in T_f} P(t|T_f,h) \log P(t|T_f,h) \tag{4.53}$$

$$H_{(\overline{T_f}|h)} \ = \ -\sum_{t \in \overline{T_f}} P(t|\overline{T_f},h) \log P(t|\overline{T_f},h) \tag{4.54}$$

ところが，図 4.6 の (b) において，最大エントロピー法の原理から，T_f に属する全ての目標事象について $P(t|T_f,h)$ の値は等しく $1/|T_f|$ となり，また，$\overline{T_f}$ に属する全ての目標事象について $P(t|\overline{T_f},h)$

の値は等しく $1/|\overline{T_f}|$ となる．したがって，式 (4.53)，式 (4.54) は以下のように簡略できる．

$$H_{(T_f|h)} = -\sum_{t\in T_f}\frac{1}{|T_f|}\log\frac{1}{|T_f|} = -\log\frac{1}{|T_f|} \tag{4.55}$$

$$H_{(\overline{T_f}|h)} = -\sum_{t\in \overline{T_f}}\frac{1}{|\overline{T_f}|}\log\frac{1}{|\overline{T_f}|} = -\log\frac{1}{|\overline{T_f}|} \tag{4.56}$$

- $H'_{(T_f,\overline{T_f})}$
  履歴事象が h であるという条件の下で，その目標事象として T_f に属する事象が現われるか $\overline{T_f}$ に属する事象が現われるかを予測する確率モデルのエントロピーである．

$$H'_{(T_f,\overline{T_f})} = -P(T_f|h)\log P(T_f|h) - P(\overline{T_f}|h)\log P(\overline{T_f}|h) \tag{4.57}$$

  確率 $P(T_f|h)$, $P(\overline{T_f}|h)$ の値は式 (4.51), 式 (4.52) により計算することができるため，式 (4.57) の値もまた計算することができる．

以上から，H_f に属する履歴事象 h についてのエントロピー $H_{(T_f|h)}$ の値は，式 (4.45), (4.51), (4.52), (4.55), (4.56), (4.57) をもとに，訓練データにおける確率分布 $\hat{P}(t,h)$ と $|T_f|$, $|\overline{T_f}|$ から計算することができる．また，$H_{(T_f|h)}$ の値は H_f に属する全ての履歴事象 h について等しいので，式 (4.43) は以下のように簡略できる．

$$H(P_{\{f,f^T\}}) = \sum_{h\in H_f}\hat{P}(h)H_{(T|h)} - \hat{P}(\overline{H_f})\log\frac{1}{|T|} \tag{4.58}$$

$$= \hat{P}(H_f)H_{(T|h)} - \hat{P}(\overline{H_f})\log\frac{1}{|T|} \tag{4.59}$$

式 (4.37) と式 (4.59) の差である $U_T(f)$ の値を計算する際には，それぞれの式の第 2 項の計算を省略できるため，$U_T(f)$ は最終的に式 (4.62) によって求められる．

$$U_T(f) = H(P_{\{f^T\}}) - H(P_{\{f,f^T\}}) \tag{4.60}$$

$$= -\hat{P}(H_f)\log\frac{1}{|T|} - \hat{P}(H_f)H_{(T|h)} \tag{4.61}$$

$$= -\hat{P}(H_f)\left\{\log\frac{1}{|T|} + H_{(T|h)}\right\} \tag{4.62}$$

このように，$U_T(f)$ の値は，訓練データにおける確率分布 $\hat{P}(t,h)$ と $|T|$, $|T_f|$, $|\overline{T_f}|$ から直ちに計算することができるため，その計算量は極めて小さい．

4.4.2 $U_H(f)$ の計算方法

$U_H(f)$ は，式 (4.30) に示した通り，素性集合 F として f^H のみを考えたとき (図 4.8 (a)) の確率モデル全体のエントロピー $H(P_{\{f^H\}})$ と，素性集合 F として f と f^H の 2 つを考えたとき (図 4.8 (b)) の確率モデル全体のエントロピー $H(P_{\{f,f^H\}})$ との差である．

まず，エントロピー $H(P_{\{f^H\}})$ の計算方法を説明する．

$$H(P_{\{f^H\}}) = -\sum_{h\in H}\hat{P}(h)\sum_{t\in T}P(t|h)\log P(t|h) \tag{4.63}$$

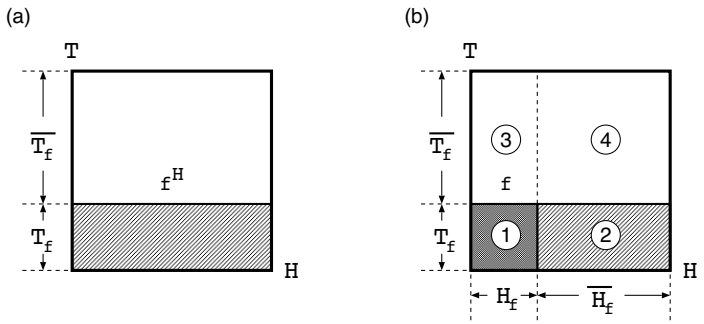


図 4.8: $U_f(H)$ の計算

$$
\begin{aligned}
&= \sum_{h \in H} \hat{P}(h)\ H_{(T|h)} && (4.64)\\
&= H_{(T|h)} && (4.65)\\
&= P(T_f|h) H_{(T_f|h)} + P(\overline{T_f}|h) H_{(\overline{T_f}|h)} + H'_{(T_f, \overline{T_f})} && (4.66)
\end{aligned}
$$

式 (4.64) から式 (4.65) の変形は，全ての履歴事象 h について $H_{(T|h)}$ の値は等しいことから導かれる．また，式 (4.66) の各項は，4.4.1 項で述べた方法と同じように，式 (4.51), (4.52), (4.55), (4.56), (4.57) から計算することができる．

次に，エントロピー $H(P_{\{f,f^H\}})$ の計算方法を説明する．

$$
\begin{aligned}
H(P_{\{f,f^H\}}) &= -\sum_{h \in H} \hat{P}(h) \sum_{t \in T} P(t|h) \log P(t|h) && (4.67)\\
&= \sum_{h \in H_f} \hat{P}(h)\ H_{(T|h)} + \sum_{h \in \overline{H_f}} \hat{P}(h)\ H_{(T|h)} && (4.68)\\
&= \hat{P}(H_f) H^1_{(T|h)} + \hat{P}(\overline{H_f}) H^2_{(T|h)} && (4.69)
\end{aligned}
$$

ここで，$H^1_{(T|h)}$ は H_f に属する履歴事象 h についての条件付き確率 $P(t|h)$ のエントロピー，$H^2_{(T|h)}$ は $\overline{H_f}$ に属する履歴事象 h についての条件付き確率 $P(t|h)$ のエントロピーを表わす．4.4.1 項で述べたように，$H^1_{(T|h)}$ および $H^2_{(T|h)}$ を求めるためには式 (4.51), (4.52), (4.55), (4.56), (4.57) を計算すればよく，これらは $|T_f|$, $|\overline{T_f}|$ と確率比 $P(T_f|h) : P(\overline{T_f}|h)$ から求めることができる．以下，$h \in H_f$ および $h \in \overline{H_f}$ のそれぞれの場合について，確率比 $P(T_f|h) : P(\overline{T_f}|h)$ を計算する方法について説明する．

最大エントロピー法の原理から，図 4.8 の ①，②，③+④ で示した事象集合の推定された確率モデルにおける確率和は訓練データにおける確率和と等しい．

$$
\begin{aligned}
P(T_f, H_f) &= \hat{P}(T_f, H_f) && (4.70)\\
P(T_f, \overline{H_f}) &= \hat{P}(T_f, \overline{H_f}) && (4.71)\\
P(\overline{T_f}, H) &= \hat{P}(\overline{T_f}, H) && (4.72)
\end{aligned}
$$

$$
\text{但し，} \quad P(X,Y) = \sum_{t \in X, h \in Y} P(t,h) \quad , \quad \hat{P}(X,Y) = \sum_{t \in X, h \in Y} \hat{P}(t,h) \qquad (4.73)
$$

さらに，③+④ で示した事象集合の個々の事象の同時確率 $P(t,h)$ は全て等しいので，次式が成立する．

$$
P(\overline{T_f}, H_f) = \frac{|H_f|}{|H|} P(\overline{T_f}, H) = \frac{|H_f|}{|H|} \sum_{t \in \overline{T_f}, h \in H} \hat{P}(t,h) \qquad (4.74)
$$

$$P(\overline{T_f},\overline{H_f}) \;=\; \frac{|\overline{H_f}|}{|H|}\,P(\overline{T_f},H) \;=\; \frac{|\overline{H_f}|}{|H|}\sum_{t\in\overline{T_f},h\in H}\hat{P}(t,h) \tag{4.75}$$

また，$h \in H_f$ であるときの確率比 $P(T_f|h)\ :\ P(\overline{T_f}|h)$ は，図 4.8 における事象集合 ① と ③ の確率和の比に等しく，$h \in \overline{H_f}$ であるときの確率比 $P(T_f|h)\ :\ P(\overline{T_f}|h)$ は，図 4.8 における事象集合 ② と ④ の確率和の比に等しい．

$$h \in H_f\text{のとき} \qquad P(T_f|h)\ :\ P(\overline{T_f}|h) \;=\; P(T_f,H_f)\ :\ P(\overline{T_f},H_f) \tag{4.76}$$

$$h \in \overline{H_f}\text{のとき} \qquad P(T_f|h)\ :\ P(\overline{T_f}|h) \;=\; P(T_f,\overline{H_f})\ :\ P(\overline{T_f},\overline{H_f}) \tag{4.77}$$

式 (4.70), (4.71), (4.74), (4.75), (4.76), (4.77) により，$h \in H_f$ および $h \in \overline{H_f}$ のときの確率比 $P(T_f|h)$: $P(\overline{T_f}|h)$ の値は訓練データにおける確率分布 $\hat{P}(t,h)$ と $|H|$, $|H_f|$, $|\overline{H_f}|$ から求めることができる．したがって，式 (4.69) における $H^1_{(T_f|h)}$, $H^2_{(T_f|h)}$ の値も計算できる．

以上述べたように，式 (4.66) および式 (4.69) の値は，訓練データにおける確率分布 $\hat{P}(t,h)$ と $|T|$, $|T_f|$, $|\overline{T_f}|$, $|H|$, $|H_f|$, $|\overline{H_f}|$ から直ちに計算することができるため，$U_H(f)$ の計算量は極めて小さい．

4.5 評価実験

本節では，最大エントロピー法を適用して確率モデルを推定する実験について説明する．また，4.4 節で提案した素性効用に基づく素性選択と素性選択アルゴリズム [5] との比較実験について述べる．

本節の実験で推定する確率モデルは，統合的確率言語モデルにおける動詞に係る助詞の生成確率 (式 (3.74)) である．この確率モデルを以下に再掲する[4]．

$$P(\vec{p}|v,n) \;=\; P(\vec{p}|c_v,cf_v,syn_v,n) \qquad where\ n=2\ or\ 3 \tag{4.78}$$

式 (4.78) は，n 個の助詞がある動詞 v に同時に係っているときに，それらから n 個の単語 $\vec{p}=(p_1,\cdots,p_n)$ が生成される確率である．また，3.2.2.2 で述べたように，c_v は動詞の意味クラス，cf_v は動詞の格フレーム，syn_v は動詞の統語的特性である．本実験では，$n=2$ または $n=3$ のときの確率モデルを個別に学習する．

$$n=2\text{ のとき} \qquad P(\vec{p}|\vec{v}) = P(p_1,p_2|c_v,cf_v,syn_v) \tag{4.79}$$

$$n=3\text{ のとき} \qquad P(\vec{p}|\vec{v}) = P(p_1,p_2,p_3|c_v,cf_v,syn_v) \tag{4.80}$$

これらの確率モデルにおいて，目標事象 t は $\vec{p}=(p_1,p_2)$ もしくは $\vec{p}=(p_1,p_2,p_3)$ であり，履歴事象 h は $\vec{v}=(c_v,cf_v,syn_v)$ である．これらの確率モデルを以下の手順により推定した．

1. 表 3.3 に示した事例集合を基に，訓練共起データ $O(\vec{p},\vec{v})$ を作成する．
2. 素性候補の集合 S をつくる．
3. S から確率モデルの推定に有効な素性を選択し，素性集合 F とする．これを以下の 2 つの方法を用いて行う．
   - 素性効用に基づく素性選択 (4.4 節)
   - 効率化された素性選択アルゴリズム (4.3 節)
4. 素性集合 F と訓練共起データから，効率化された GIS アルゴリズム (4.2 節) を用いて確率モデルを推定する．

[4] 簡単のため，$\vec{p'}$ を $\vec{p}$ と表記する．

4.5.1 素性候補の集合 S の作成

式 (4.79)，式 (4.80) の確率モデルの推定に用いる素性として，以下に示す f^1〜f^{12} の 12 種類の素性を考えた．

1.
$$f^1_{(P)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; P \in \vec{p} \\ 0 & otherwise \end{cases} \tag{4.81}$$

助詞の生成確率を考慮する素性である．例えば，訓練データにおいて助詞「を」がよく出現する場合，$f^1_{(\text{を})}$ という素性を与えることにより，式 (4.6) の制約から $\vec{p}$ に「を」が現われるときの確率が高く推定される．

2.
$$f^2_{(P,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; P \in \vec{p} \;\; \& \; SYN_v = syn_v \\ 0 & otherwise \end{cases} \tag{4.82}$$

ある特定の統語的特性 SYN_v について助詞の生成確率を考慮する素性である．例えば，訓練データにおいて，動詞の統語的特性が “主辞” である場合に限り助詞「は」がよく出現する場合，$f^2_{(\text{は, 主辞})}$ という素性を与えることにより，式 (4.6) の制約から $\vec{p}$ に「は」が現われるときの確率が高く推定される．

3.
$$f^3_{(P_1,P_2)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; (P_1,P_2) \subset \vec{p} \\ 0 & otherwise \end{cases} \tag{4.83}$$

2 つの助詞の従属関係を考慮する素性である．例えば，訓練データにおいて「が」と「に」が同時に出現することが多く観察される場合，$f^3_{(\text{が, に})}$ という素性を与えることにより，式 (4.6) の制約から $\vec{p}$ に「が」と「に」が同時に現われるときの確率が高く推定される．また，動詞が二重格を持つ事例の出現頻度が低い場合には，$f^3_{(\text{が, が})}$ のような素性を与えることにより，$\vec{p}$ に同じ助詞が含まれるときの確率が低く推定される．

素性 (4.83) において，“$(P_1,P_2) \subset \vec{p}$” は，2 つの助詞 P_1 と P_2 が $\vec{p}$ の中にこの順序で現われることを示している．例えば，(が, を) $\subset$ (が, と, を) は成立するが，(が, を) $\subset$ (を, と, が) は成立しない．すなわち，助詞が現われる順番も考慮されている．

4.
$$f^4_{(P_1,P_2,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; (P_1,P_2) \subset \vec{p} \;\; \& \; SYN_v = syn_v \\ 0 & otherwise \end{cases} \tag{4.84}$$

ある特定の統語的特性 SYN_v について 2 つの助詞の従属関係を考慮する素性である．

5.
$$f^5_{(P,C_v,CF_v,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; P \in \vec{p} \;\; \& \; (C_v,CF_v,SYN_v) = \vec{v} \\ 0 & otherwise \end{cases} \tag{4.85}$$

ある特定の履歴事象 $\vec{V} = (C_v, CF_v, SYN_v)$ に限り，目標事象としてある助詞 P がよく現われる，もしくは現われにくい場合，それを訓練データから学習するための素性である．

6.
$$f^6_{(P,C'_v,CF_v,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if \;\; P \in \vec{p} \;\; \& \; C'_v \succ c_v \;\; \& \; CF_v = cf_v \;\; \& \; SYN_v = syn_v \\ 0 & otherwise \end{cases} \tag{4.86}$$

素性 $f^5_{(P,C_v,CF_v,SYN_v)}$ のうち，動詞クラス C_v をより抽象度の高い意味クラス C'_v に置き換えた素性である．素性 (4.86) における “$C'_v \succ c_v$” は，c_v が C'_v の下位に属する意味クラスであることを意味する．

今回の実験では抽象度の高い意味クラス C'_v として，分類語彙表における 2 桁の分類コードを利用した．尚，3.2.2.2 で述べたように，c_v は分類語彙表における 3 桁の分類コードである．

7.
$$f^7_{(P,C_v,CF_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ P \in \vec{p}\ \ \&\ C_v = c_v\ \ \&\ CF_v = cf_v \\ 0 & otherwise \end{cases} \tag{4.87}$$

素性 $f^5_{(P,C_v,CF_v,SYN_v)}$ と同様の働きを持ち，かつ動詞の統語的特性 syn_v を無視した素性である．

8.
$$f^8_{(P,C'_v,CF_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ P \in \vec{p}\ \ \&\ C'_v \succ c_v\ \ \&\ CF_v = cf_v \\ 0 & otherwise \end{cases} \tag{4.88}$$

素性 $f^6_{(P,C'_v,CF_v,SYN_v)}$ と同様の働きを持ち，かつ動詞の統語的特性 syn_v を無視した素性である．

9.
$$f^9_{(P_1,P_2,C_v,CF_v,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ (P_1,P_2) \subset \vec{p}\ \ \&\ (C_v,CF_v,SYN_v) = \vec{v} \\ 0 & otherwise \end{cases} \tag{4.89}$$

ある特定の履歴事象 $\vec{V} = (C_v, CF_v, SYN_v)$ に限り，目標事象としてある助詞 P_1 と P_2 が同時によく現われる，もしくは現われにくい場合，それを訓練データから学習するための素性である．

10.
$$f^{10}_{(P_1,P_2,C'_v,CF_v,SYN_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ (P_1,P_2) \subset \vec{p}\ \ \&\ C'_v \succ c_v\ \ \&\ CF_v = cf_v\ \ \&\ SYN_v = syn_v \\ 0 & otherwise \end{cases} \tag{4.90}$$

素性 $f^9_{(P_1,P_2,C_v,CF_v,SYN_v)}$ のうち，動詞クラス C_v をより抽象度の高い意味クラス C'_v に置き換えた素性である．素性 $f^6_{(P,C'_v,CF_v,SYN_v)}$ と同様に，C'_v として分類語彙表における 2 桁の分類コードを利用した．

11.
$$f^{11}_{(P_1,P_2,C_v,CF_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ (P_1,P_2) \subset \vec{p}\ \ \&\ C_v = c_v\ \ \&\ CF_v = cf_v \\ 0 & otherwise \end{cases} \tag{4.91}$$

素性 $f^9_{(P_1,P_2,C_v,CF_v,SYN_v)}$ と同様の働きを持ち，かつ動詞の統語的特性 syn_v を無視した素性である．

12.
$$f^{12}_{(P_1,P_2,C'_v,CF_v)}(\vec{p},\vec{v}) = \begin{cases} 1 & if\ \ (P_1,P_2) \subset \vec{p}\ \ \&\ C'_v \succ c_v\ \ \&\ CF_v = cf_v \\ 0 & otherwise \end{cases} \tag{4.92}$$

素性 $f^{10}_{(P_1,P_2,C'_v,CF_v,SYN_v)}$ と同様の働きを持ち，かつ動詞の統語的特性 syn_v を無視した素性である．

助詞 P，動詞の意味クラス C_v，C'_v，動詞の格フレーム CF_v，動詞の統語的特性 SYN_v の全ての組み合わせについて，素性 f^1〜f^{12} の全てを素性候補の集合 S とするのは効率が悪い．そこで，素性が 1 を返す履歴事象の訓練データにおける出現回数がある一定の閾値 λ 以上の場合に限り，それらを素性候補の集合 S の要素とした．例えば，素性 $f^5_{(P,C_v,CF_v,SYN_v)}$ の場合，事象 $(*, (C_v, CF_v, SYN_v))$[5]の出現回数が λ 以上の場合に限り，その素性を S の要素とした．同様に，素性 $f^7_{(P,C_v,CF_v)}$ の場合，事象 $(*, (C_v, CF_v, *))$[6]の出現回数が λ 以上の場合に限り，その素性を S の要素とした．

今回の実験に使用した素性候補の集合 S の概要を表 4.1 に示す．

表 4.1 に示したように，素性 f^1〜f^4，f^5〜f^8，f^9〜f^{12} の種類によって閾値 λ を変化させている．これは素性の一般性を考慮したものである．素性の一般性とは，簡単に言えば素性が 1 を返す事象集合の大きさ (図 4.2 における長方形の面積) であり，これが大きい素性ほどより一般的であるとみなす．例えば，目

[5] $*$ は任意の $\vec{p}$ を表わす．
[6] $*$ は任意の $\vec{p}$ もしくは syn_v を表わす．

表 4.1: 素性候補の集合 S の概要

※ $n = 2$ のとき

	f^1〜f^4	f^5〜f^8	f^9〜f^{12}	合計
λ	0	200	2,000	—
素性候補数	240	3915	12,633	16,788

※ $n = 3$ のとき

	f^1〜f^4	f^5〜f^8	f^9〜f^{12}	合計
λ	0	50	500	—
素性候補数	240	3615	12,838	16,693

標事象に対する条件が満たされればいかなる履歴事象に対しても 1 を返す素性 f^1〜f^4 は，素性 f^5〜f^8 や f^9〜f^{12} よりも一般性が高い．また，2 つの助詞を目標事象に含むか否かを条件とする素性 f^9〜f^{12} は，1 つの助詞を目標事象に含むか否かを条件とする素性 f^5〜f^8 よりも一般性が低い．素性の種類の一般性が低ければ低いほど，すなわち 1 を返す事象集合の大きさが小さければ小さいほど，その素性の種類に分類される素性候補の数は多くなる．より一般性の低い素性の種類に対して閾値 λ の値を大きく設定したのはこのためである．ちなみに，表 4.1 において，素性 f^1〜f^4 に対する閾値 λ が 0 となっているのは，全ての助詞 P，統語的特性 SYN_v の組み合わせについて，それに対応する全ての素性 f^1〜f^4 を素性候補の集合 S の要素としたことを意味する．

4.5.2 確率モデル推定

4.5.1 項で作成した素性候補の集合 S から，以下の手順に従って素性集合 F を選択した．

- 素性効用に基づく素性選択
  1. あらゆる助詞 P および SYN_v について，素性 $f^1_{(P)}$ および $f^2_{(P,SYN_v)}$ は確率モデルの学習に有効であるとみなして，F の要素として追加した．
  2. あらゆる助詞 P および SYN_v について，二重格に関する素性 $f^3_{(P,P)}$ および $f^4_{(P,P,SYN_v)}$ は，同様に確率モデルの学習に有効であるとみなして，F の要素として追加した．
  3. 残りの素性候補について，式 (4.28) で定義される素性効用を計算する．そして，1. および 2. で選択した素性と合わせて，F の要素数が F_n となるように，効用の大きい素性を F の要素として追加した．

- 素性選択アルゴリズムによる素性選択
  1. 素性候補の集合 S から，4.3 節で述べた効率化した素性選択アルゴリズムにより素性集合 F を選択した．ここでは，一度に選択する素性の数 N_f を 20 として実験を行った．

次に，このようにして得られた素性集合 F と訓練データから，4.2 節で述べた効率化した GIS アルゴリズムによって確率モデル $P(\vec{p}|\vec{v})$ を推定した．結果を表 4.2 に示す．

表 4.2 においては，素性効用に基づく素性選択，および素性選択アルゴリズムの評価尺度として，以下の 3 つを用いた．

表 4.2: 最大エントロピー法による確率モデル推定実験の結果

※ $n = 2$ のとき

	素性効用による素性選択			素性選択アルゴリズム
	$F_n = 1000$	$F_n = 2000$	$F_n = 3000$	$F_n = 1000$
TP	3.67	3.63	3.61	3.58
CPU 時間 [sec]	21			31,854
実時間 [sec]	1,155			31,916

※ $n = 3$ のとき

	素性効用による素性選択			素性選択アルゴリズム
	$F_n = 1000$	$F_n = 2000$	$F_n = 3000$	$F_n = 500$
TP	5.72	5.67	5.64	5.39
CPU 時間 [sec]	212			79,472
実時間 [sec]	1,137			79,843

- テストセットパープレキシティ TP

$$TP = -\sum_{\vec{v} \in Test\,Set} \tilde{P}(\vec{v}) \times \sum_{(\vec{p},\vec{v}) \in Test\,Set} \tilde{P}(\vec{p}|\vec{v}) \cdot \log P(\vec{p}|\vec{v}) \tag{4.93}$$

  式 (4.93) において，$\tilde{P}(\vec{v})$ および $\tilde{P}(\vec{p}|\vec{v})$ は，与えられたテストセットの集合 $(\vec{p}, \vec{v})$ から最尤推定した確率を表わす．今回の実験では，京大コーパス 8924 文から抽出された事象 $(\vec{p}, \vec{v})$ の集合をテストセットとした．テストセットとした事象の数は，$n = 2$ のときで 3,334 組，$n = 3$ のときで 1,606 組である．推定した確率モデル $P(\vec{p}|\vec{v})$ がテストセットに現われた事象に対して高い確率を与えれば与えるほど，この TP の値は小さくなる．したがって，TP の値が小さければ小さいほど，より良い確率モデルが推定されたとみなすことができる．

- CPU 時間

  CPU が素性選択に要した計算時間．単位は秒である．

- 実時間

  ディスクの IO 時間などを含めて，素性選択の開始から終了までに要した時間．単位は秒である．

表 4.2 から，本研究で提案した素性効用に基づく素性選択の場合，素性選択アルゴリズムに比べて，素性集合 F の選択に要する時間が CPU 時間で 300 分の 1 以下，実時間で 25 分の 1 以下となっていることがわかる．素性効用に基づく素性選択で，CPU 時間と実時間に大きな差があるのは，ディスク IO，特に素性候補の集合 S の読み込みに時間がかかっているためと思われる．また，素性効用は全ての素性候補についてただ一度だけ計算すればよいので，選択する素性の数 F_n が多くなっても，ほぼ同じ時間で素性を選択することができる．一方，素性選択アルゴリズムは，$n = 2$ で 1000 個，$n = 3$ のときで 500 個の素性を選択するのにそれぞれ実時間で約 9 時間，約 22 時間の時間を要する．さらに多くの素性を選択しようとする場合，もしくはさらにパラメタ数の多い確率モデルを推定しようとする場合には，それに要する計算時間は無視できないほど大きくなる可能性がある．

一方，素性効用に基づく素性選択によって推定された確率モデルの TP は，確率モデルの推定に用いる素性の数 F_n を $1000, 2000, 3000$ と増やすにつれて，その値も小さくなっていることがわかる．また，素性効用に基づく素性選択によって推定された確率モデルの TP は，素性選択アルゴリズムによるものと比べて，わずかに大きくなっている．これは，素性選択アルゴリズムが素性を順番に素性集合 F に加えていき，既に加えられた素性とそうでない素性の依存関係を考慮して素性選択を行っているのに対し，素性効用に基づく素性選択の場合は他の素性との依存関係を考慮していないためと考えられる．素性が確率モデルの推定に有効かどうかは，他の素性が確率モデルに加えられているかどうかにも依存する．このような素性間の依存関係を考慮し，かつ計算量をなるべく少なく抑えて素性選択を行う方法も今後検討していく必要がある．

最後に，推定した確率モデル $P(\vec{p}|\vec{v})$ を 3 章で提案した統合的確率言語モデルの一部として加えたときの文節の正解率を表 4.3 に示す．

表 4.3: $P(\vec{p}|\vec{v})$ を統合的確率言語モデルに組み込んだときの文節の正解率

	素性効用による素性選択	素性選択アルゴリズム
文節の正解率	82.7%	82.6%

この実験設定の詳細は表 3.5 に示したモデル “all” とほぼ同じである[7]．両者の文節の正解率はほとんど変わらなかった．したがって，動詞に係る助詞の生成確率 $P(\vec{p}|\vec{v})$ を推定する場合には，4.4 節で提案した素性効用に基づいて素性選択を行っても，素性選択アルゴリズムを用いた場合と同じ程度の品質の確率モデルを推定できることがわかる．尚，3.2 節の実験においては，式 (3.74) を最大エントロピー法を用いて推定する際には，素性効用に基づいて素性集合 F を選択している．

[7] 3.2 節の実験では，$\vec{p}$ の要素として現われる助詞の数を 17 個に限定していた．本節の実験では，素性選択アルゴリズムの計算量が非常に大きいため，$\vec{p}$ の要素として現われる助詞の数を 14 個に限定して実験を行っている．文節の正解率が表 3.5 で示したものと異なるのはこのためである．

第5章　文脈自由文法の自動獲得

前章まででは，言語資源から学習された統計情報を利用した曖昧性解消について主に論じた．しかしながら，自然言語文を解析する際には，曖昧性解消に用いる統計情報だけでなく，辞書や文法などといった自然言語処理用知識が必要である．この自然言語処理用知識を言語資源から自動的に獲得する研究は，統計情報を利用した曖昧性解消と並んで，統計的自然言語処理における重要な応用例のひとつである．1 章で述べたように，大量の言語資源から自動的に獲得した知識は，人手によって得られる知識と比べて，獲得した知識が人間の主観に影響されにくい，知識作成のためのコストが低い，知識の適用範囲が広い，といった優れた特長を持っている．特に，多様な自然言語文を解析するためには，解析に用いる自然言語処理用知識が様々な言語現象に対応していなければならないという問題があるが，この問題を解決するひとつの方法として，言語資源から自動獲得された適用範囲の広い知識を用いることが挙げられる．

言語資源から自動獲得される自然言語処理用知識には様々なものがあるが，その中の 1 つとして CFG が挙げられる．CFG を人手によって作成する場合，作成の際に考慮されなかった言語現象については，それに対応する規則が CFG に含まれていないために解析することができないという問題点がある．これに対して，コーパスから自動的に CFG を獲得することができれば，コーパス内に現れる多様な言語現象を網羅できるだけでなく，人的負担も極めて軽くなる．

本章では，CFG をコーパスから自動的に獲得する手法を提案する．まず，5.1 節では，コーパスから CFG およびそれに相当する文法を獲得する過去の研究について概観し，それらの問題点について述べる．5.2 節では，本研究で提案する CFG を自動獲得する手法を詳説する．本手法の特徴は，文法獲得に要する計算量が既存研究に比べて少なく，大量の言語資源に適用することによって多様な自然言語文を受理する CFG を獲得することができる点にある．5.3 節では，5.2 節で述べた手法によって獲得された CFG を洗練するいくつかの手法を提案する．最後に 5.4 節では，実際にコーパスから CFG を自動獲得し，それを用いた構文解析実験を行うことにより，獲得された CFG の品質を評価する．

5.1　文法獲得に関する既存研究

本節では，CFG もしくは PCFG をコーパスから自動獲得する過去の研究について述べる．

文法獲得に利用されるコーパスとしては次のような種類がある．

- 平文コーパス

  例文のみを収集したコーパス．

- 品詞タグ付きコーパス

  例文の各単語に品詞が与えられたコーパス．日本語の場合には形態素解析がなされたコーパス，すなわち単語の区切りと品詞が与えられたコーパスを指す．

- 括弧付きコーパス

  各例文に括弧付けがなされたコーパス．括弧付けとは，例文中の統語的なまとまりを括弧でくくることによりその文の構造を示したものである．括弧付けの例を以下に示す．

[[よく [[山 に] 登る] 。]

「山」と「に」をくくった括弧は「山に」が統語的なまとまりであることを示し,「山に」と「登る」をくくった括弧は「山に登る」が統語的なまとまりであることを示す．この括弧付けは内部ノードに構成素ラベルのない構文木 (unlabeled tree) と等価である．両者の対応を図 5.1 に示す．

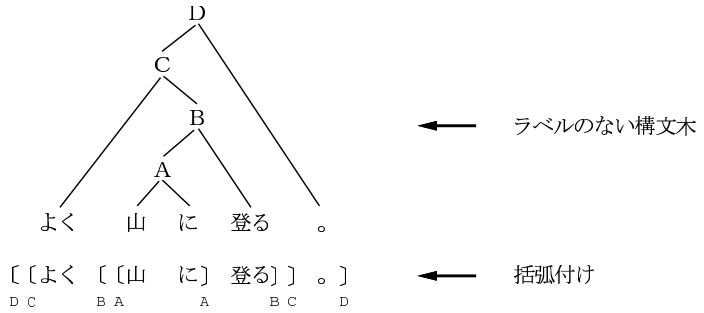


図 5.1: 括弧付けと構文木の対応

図 5.1 の A,B,C,D はノードの識別子であって構成素ラベルではない．また，各ノードが支配する単語列が 1 組の括弧によってくくられている．図 5.1 中の括弧の下のアルファベットはノードと括弧付けの対応を示したものである．以下，括弧付けに対応した構成素ラベルのない構文木を**括弧付けによる構文構造**と呼ぶ．

- 構文木付きコーパス

  各例文に構文木 (labeled tree) が与えられたコーパス．構文木の内部ノードに “名詞句” などの構成素ラベルが与えられている点が括弧付きコーパスと異なる．

以下，文法獲得に関する過去の研究が，どのような種類のコーパスからどのような手法を用いて行われているのかについて概観する．

5.1.1 平文コーパスからの文法獲得

平文コーパスからの文法規則獲得に関する研究としては Kiyono らによるものがある [39, 40, 41]．Kiyono らの方法は，まずコーパスの文を初期の CFG を用いて構文解析し，解析に失敗した際に生成された部分木から，解析に失敗した文の構文解析を成功させるために必要な規則 (彼らは仮説と呼んでいる) を見つけ出す．次に，その仮説がコーパスの文の解析を成功させるのにどの程度必要なのかを表わす尤度 (Plausibility) を計算し，高い尤度を持つ仮説を新たな規則として文法に加える．Kiyono らは全ての文法規則を獲得することを目的としているわけではなく，初期知識としてある程度正しい CFG を用意し，それを新たな領域に適用する際にその領域に固有の言語現象を取り扱うために必要な規則を自動的に獲得することを目的としている．

Kupiec らは本来品詞タグ付きコーパスを必要とする Inside-Outside アルゴリズムを平文コーパスに適用する方法を提案している [45]．彼らの手法は，まず単語の品詞付けを行う tagger を獲得し，その tagger を用いて平文コーパスに品詞付けを行った後，Inside-Outside アルゴリズムによって文法を獲得するというものである．tagger は基本的には HMM を用いて学習するが，学習するパラメタの数を減らすために語の等

価クラスという概念を導入したり，接尾から未知語の品詞を推定するなどの改良を加えている [44]．この手法の長所は平文コーパスを利用して文法を獲得するため，品詞集合の変化にすぐ対応できるなどの柔軟性に優れている点にある．

5.1.2 品詞タグ付きコーパスからの文法獲得

品詞タグ付きコーパスから CFG を獲得する研究としては森らによるものがある [62]．森らは，前後に現われる品詞に無関係に出現する品詞列を独立度の高い品詞列と定義し，コーパスに現われる品詞列の独立度を n-gram 統計により評価する．次に，ある一定の閾値以上の独立度を持つ品詞列を規則の右辺として取り出す．また，取り出された品詞列の集合に対して，その前後に現われる品詞の分布傾向を利用してクラスタリングを行い，同一クラスタと判断された品詞列を右辺とする規則の左辺に同一の非終端記号を与える．そして，得られた規則のクラスタの中からコーパス中によく現れるものを選び，それらを CFG 規則として採用すると同時に，コーパス中に現われる規則の右辺の品詞列を左辺の非終端記号に置き換える．このような操作を繰り返すことにより，最終的な CFG を獲得すると同時に，コーパスの各例文に構文木を付加している．

また，Inside-Outside アルゴリズムを利用して CFG を獲得する研究も報告されている [48]．Inside-Outside アルゴリズムとは品詞タグ付きコーパスから PCFG の規則の確率を学習するアルゴリズムであり，HMM のパラメタ学習アルゴリズムである Forward-Backward アルゴリズム [69] を拡張したものである．Inside-Outside アルゴリズムによる PCFG の規則の確率の学習は以下のように行われる．

1. 規則の初期確率をランダムに与える．
2. PCFG を用いてコーパスの文を構文解析し，得られた全ての解析木の生成に用いられた規則の出現回数を数え，それを元に各規則の確率の再推定を行う．
3. 規則の確率が収束するまで 2. の操作を反復する．

このように，Inside-Outside アルゴリズムはもともと規則の確率を学習するためのアルゴリズムであるが，Lari らはこれを文法獲得に応用する方法も提案している [48]．彼らの手法は，与えられた終端記号と非終端記号の集合からそれらを組み合わせてできる全てのチョムスキー標準形の CFG 規則を作り，それらの確率を Inside-Outside アルゴリズムによって学習し，確率の低い規則を削除することにより新たな PCFG を獲得するというものである．ここで，CFG がチョムスキー標準形であるとは，規則の形式が次の 2 つのいずれかに限られることを指す．

$$\begin{array}{lcl} A & \rightarrow & B\ C \\ A & \rightarrow & w \end{array}$$

但し，A, B, C は非終端記号を，w は終端記号を表わす．

5.1.3 括弧付きコーパスからの文法獲得

5.1.2 項で述べた Inside-Outside アルゴリズムを用いた文法獲得は，収束性が悪い，計算量が多いなどの問題点も多い．これに対して，Pereira らは部分的に括弧付けされたコーパスに対して Inside-Outside アルゴリズムを適用する方法を提案した [68]．Pereira らの手法では，5.1.2 項で述べた Inside-Outside アルゴリズムの Step 2. において，コーパスの括弧付けと矛盾するような解析木 (つまり正しくない解析木) を無視することにより，矛盾する解析木に現れた規則の出現回数を規則の確率の再推定に使用しない．その結果，Inside-Outside アルゴリズムの収束性および得られた PCFG を用いた構文解析の精度が向上したと

報告している．また，完全に括弧付けがなされた[1] コーパスを使用すれば，計算量を $O(n^3)$ から $O(n)$ (n は訓練コーパスの文の長さ) に削減できることを証明している．Schabes らは，Pereira らと同じ方法を用いて Inside-Outside アルゴリズムを大規模な Wall Street Journal コーパスに適用している [84]．彼らは Inside-Outside アルゴリズムに要する計算量を減少させるために，部分的に括弧付けされたコーパスを完全に括弧付けされたコーパスに自動的に変換する工夫を施している．しかしながら，Inside-Outside アルゴリズムを利用して PCFG の獲得を行う場合には，局所解は得られるが最適解が得られる保証はないといった問題点も残されている．

Sakakibara は上昇型統語解析器に適合した reversible grammar と呼ばれる文法を括弧付きコーパスから獲得する方法を提案している [79, 80]．reversible grammar とは次の 2 つの条件を満す CFG である．

1. $A \rightarrow \alpha$ かつ $B \rightarrow \alpha$ なら，$A = B$ である．
2. $A \rightarrow \alpha\ B\ \beta$ かつ $A \rightarrow \alpha\ C\ \beta$ なら，$B = C$ である．

Sakakibara の方法は，まずコーパスの括弧付けによる構文構造の全ての内部ノードに異なるラベルを与えて得られる文法を初期文法とする．次に，文法が 1., 2. の条件を満たすようにラベルをマージすることによって文法を獲得する．ラベルをマージする条件は以下の通りである．

- 同じ子供を持つノードが 2 つあればそのノードのラベルをマージする．すなわち，次の 2 つの規則があれば A と A' をマージする．

$$\begin{array}{lcl} A & \rightarrow & A_1\ A_2\ \cdots\ A_k \\ A' & \rightarrow & A_1\ A_2\ \cdots\ A_k \end{array}$$

- 同じラベルを持つ 2 つのノードが 1 つを除いて同じ子供を持つ場合，その 1 つのラベルをマージする．すなわち，次の 2 つの規則があれば A_i と A'_i をマージする．

$$\begin{array}{lcl} A & \rightarrow & A_1\ A_2\ \cdots\ A_i\ \cdots\ A_k \\ A & \rightarrow & A_1\ A_2\ \cdots\ A'_i\ \cdots\ A_k \end{array}$$

また Sakabibara は，文法の獲得に必要な計算量が訓練用コーパスの内部ノード数の多項式時間に比例することを証明している．

括弧付きコーパスから日本語の CFG を獲得する研究としては横田らのものがある [107, 108]．横田らは，Shift-Reduce パーザによる訓練コーパスの例文の構文解析の効率が最も良くなるように，コーパスの内部ノードに人工的な非終端記号を割り当てることにより CFG を獲得する方法を提案している．これは組み合わせ最適化問題となり，Simulated Annealing 法を用いることにより解決を求めている．1000〜7500 例文から CFG を獲得し，それを用いた構文解析では 15〜47%の正解率が得られたと報告している．

Brill は CFG を用いて構文解析を行う代わりに変形規則を用いた構文解析を考案し，その変形規則を括弧付きコーパスから学習する方法を提案している [11]．変形規則を用いた構文解析の手続きを以下に示す．

1. 入力文に対して，初期の解析木として右下がりの二分木を与える．但し，最後のピリオド “.” は一番上のレベルに置く．例えば，入力文が “The dog and old cat ate.” なら，

   [[The [dog [and [old [cat ate]]]]] .]

   という木を初期木とする．
2. 初期木に対して変形規則を適用する．変形規則の形式は以下の通り．
   - 品詞 X の { 左 | 右 } にある { 左 | 右 } 括弧を { 削除 | 追加 } せよ．

[1] ここでいう完全な括弧付けとは，括弧付けによる構文構造が完全な二分木であることをいう．

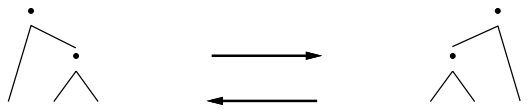


図 5.2: 変形規則による二分木の変形

- 品詞 X と Y の間にある { 左 | 右 } 括弧を { 削除 | 追加 } せよ.

変形規則の適用は図 5.2 に示した二分木の変形のどちらかの操作に等しい．また，変形規則を適用する順番はあらかじめ決められており，条件を満たす変形規則があればそれを適用する．

3. 全ての適用可能な変形規則を適用した後に残された木が構文解析結果となる．

変形規則の学習は，括弧付きコーパスの例文を変形規則を用いて構文解析することにより実現される．具体的な手順を以下に示す．

1. 訓練用コーパスを構文解析した結果の解析木として，前述の右下がりの解析木を用意する．
2. 構文解析途中の解析木に対して適用可能なあらゆる変形規則を考え，変形後の解析木が最も正しくなるようなものを 1 個選択する．このとき，変形後の解析木の正しさを評価する関数が必要となる．Brill は解析木の全ての括弧付けのうち正しい括弧付けと交差するものの割合を解析木の正しさを表す評価関数とした．
3. 2. で選択した変形規則を構文解析途中の解析木に対して適用する．
4. 2. で選択した変形規則をリストの末尾に加える．
5. 適当な数の変形規則がリストに得られるまで 2 ～ 4 の操作を繰り返す．

4. で得られたリストが学習された変形規則とその適用順序である．Brill は ATIS(Air Travel Information System) コーパスと WSJ(Wall Street Journal) コーパスを用いて実験を行い，それぞれ 64, 83 個の変形規則を獲得した．また，これを用いてテスト文を構文解析した結果，括弧付けの正解率はそれぞれ 91.12%, 88.1%となった．いずれも Inside-Outside アルゴリズムにより学習された PCFG よりも優れた結果が得られている．

5.1.4 構文木付きコーパスからの文法獲得

構文木付きコーパスから文法を獲得する研究としては，2.3 節で述べた Sekine らの手法が挙げられる [86]. Sekine らは，Penn Treebank の中から S または NP を根ノードとする部分木を自動的に抽出し，これらを CFG 規則に変換している．これは構文木付きコーパスから文法を獲得しているとみなすことができる．しかしながら，この手法の問題は，獲得される CFG 規則の数が爆発的に増大する点にある．

Bod は構文木付きコーパスをそのまま確率文法として使用する DOP(Data Oriented Parsing) を提案した [7, 8, 9, 10]．DOP とは，構文木付きコーパスに含まれる全ての部分木を基本要素とし，これらの部分木を結合して入力文に対する解析木を導出する構文解析である．コーパスにおける部分木の出現回数を基に部分木を結合する動作に対して確率を割り当てる．導出した解析木には異なる部分木を用いた導出方法が複数存在するため，それぞれの導出方法に対する解析木の生成確率の和が導出された解析木全体の生成確率となる．Bod は全ての解析木の全ての導出について生成確率を計算する代わりに，モンテカルロ技法

を用いて入力文の長さの多項式時間で最大確率の解析木を計算することを可能にした．ATIS コーパスを用いて実験した結果，Pereira ら [68] によって獲得された PCFG よりも優れた解析結果が得られたと報告している．この方法の利点としては，コーパスそのものを文法として扱うため，文法の学習を必要としないことが挙げられる．また，SLTAG [76, 82] (2.5 節) と同様に部分木を基本要素としているため PCFG では記述しにくい長距離依存性を自然に取り扱うことができる．Bod は任意の深さの部分木を基本要素とする場合と部分木の深さに制限を加えた場合とで実験し，前者の方が優れた結果が得られることを確認している．しかしながら，この手法はまだ大規模なコーパスを用いて実験されていない．コーパスサイズが大きくなればコーパス内に含まれる部分木の数も多くなるので，文法の構成要素となる部分木の数や構文解析に要する計算量が爆発的に増える可能性がある．

構文木付きコーパスにおいては，例文に付加された構文木の内部ノードに構成素ラベル (非終端記号) が割り当てられているため，通常の CFG ならば構文木の枝分れを CFG 規則とみなすことにより容易に獲得することができる．したがって，これらの研究は CFG の獲得というよりも，むしろ CFG を拡張した文法や CFG に代わる文法枠組を提案している．

5.1.5 既存手法の問題点

大量のコーパスから CFG を獲得するには，それに要する計算量が少ないことが望ましい．ところが，統語構造情報が明示されていない平文コーパスや品詞タグ付きコーパスを用いる研究においては，統語構造情報の推測に要する計算コストが大きいといった問題がある．近年では，日本においても EDR コーパスなどの大規模な括弧付きコーパスの整備が進んでおり，効率良く CFG を獲得するためにはそのような括弧付きコーパスの統語構造情報を利用することが考えられる．一方，括弧付きコーパスを用いる既存研究においては，括弧付けによる統語構造の情報を利用しているとはいえ，反復アルゴリズムを用いているために文法獲得に要する計算量は多い．n を訓練用コーパスの長さとすれば，Pereira らの方法では，Inside-Outside アルゴリズムによる規則の確率の推定に必要な計算量は $O(n)$ であり[2]，しかもこの作業を反復しなければならない [68, 84]．また Sakakibara らの手法では，文法獲得に必要な計算量が訓練用コーパスの内部ノード数，すなわち訓練用コーパスの長さ n の多項式時間に比例する [79, 80]．横田らは文法獲得に要する計算量については厳密には言及していないが，内部ノードに与える非終端記号をランダムに変化させることを繰り返す Simulated Annealing 法を用いて CFG 規則を獲得しているため，収束までにかなりの計算量を要すると考えられる [107, 108]．Brill らの手法も，1 つの変形規則を獲得するたびに訓練用コーパスの全ての解析木をチェックしなければならないため，変形規則の学習にかなりの計算量が必要となる [11]．また，Brill が提案する変形規則を用いた構文解析は決定的に行われるために，例えばある一定の信頼度 (確率) を持つ全ての解析結果の候補を出力することができないなど，柔軟性に欠けるといった問題点もある．

本研究では，括弧付きコーパスから CFG を獲得する一手法を提案する．本手法の特長は，文法を統計情報のみから獲得する過去の先行研究と異なり，人間が持つ言語学的知見を利用して文法獲得に要する計算量を $O(n)$ に抑制した点にある．次節では本研究で提案する CFG 獲得のための具体的な手法について説明する．

[2] 厳密には，コーパスに付加された構文木が完全な二分木のときのみ $O(n)$ となり，それ以外の場合の計算量は $O(n)$ よりも多い．

5.2 CFG 獲得アルゴリズム

5.2.1 EDR コーパスの概要

本研究では，文法獲得に用いる括弧付きコーパスとして EDR 日本語コーパス [67] を使用する．EDR コーパスに収録されている例文数は 201,339 である．それぞれの文には補助情報として形態素情報，構文情報，意味情報が付加されている．ここでは特に形態素情報 (品詞情報) と括弧付けによる構文構造を利用する．EDR コーパスの例文，及びそれに付加された形態素情報・括弧付けによる構文構造の例を図 5.3 に示す．

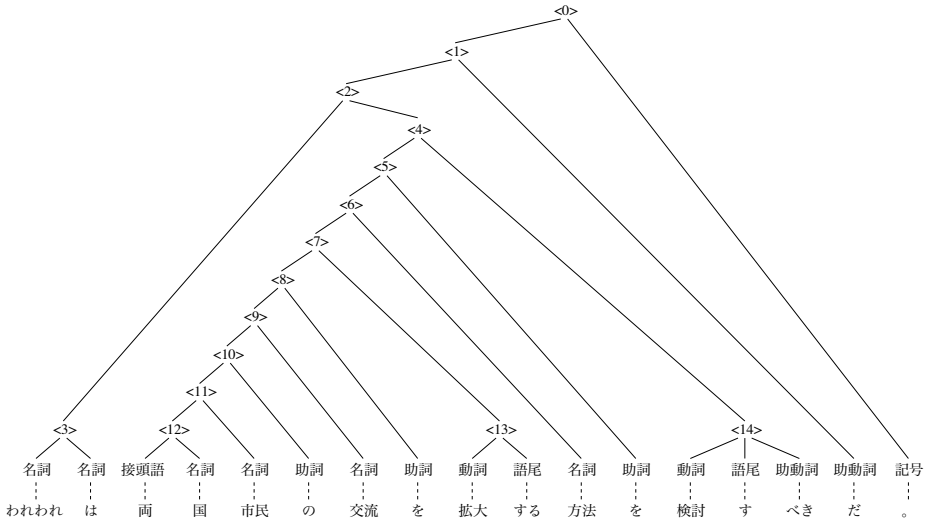


図 5.3: EDR コーパスの構文構造

EDR コーパスで使われている品詞は以下に挙げる 15 種類であり，比較的粗い品詞体系になっている．

名詞，動詞，形容詞，形容動詞，連体詞，副詞，接続詞，数字，感動詞，助詞，
助動詞，語尾，接頭語，接尾語，記号

但し，EDR コーパスにおいては，“動詞” という品詞は動詞語幹に対して割り当てられ，語尾には “語尾” という品詞が割り当てられている．同様に，“形容詞”，“形容動詞”，“助動詞” という品詞は，それぞれ形容詞語幹，形容動詞語幹，助動詞語幹に割り当てられている．

5.2.2 構文構造の内部ノードへの非終端記号の付与

図 5.3 は図 5.4 のような書き換え規則の集合とみなすことができる．これらの規則は CFG 規則と似てはいるがそうではない．なぜなら，規則中に使われている 0 から 14 までの数字は構文構造の内部ノード番号を表しているに過ぎないからである．そこで，図 5.3 のような構文構造の各ノードに対して適切なラベル (非終端記号) を割り当てることができれば，図 5.4 の規則は CFG 規則となる．このように，括弧付けによる構文構造の内部ノードに適切なラベルを与えることは括弧付きコーパスから CFG を獲得することと等価である．以下，構文構造の内部ノードに与えるラベルを決定する具体的な方法について説明する．

日本語の特徴として，前の要素が後ろの要素を修飾する，すなわち句の主辞はその句における一番最後の要素であるということが知られている [61]．例えば，図 5.4 の中の

⟨0⟩	→	⟨1⟩	記号		⟨8⟩	→	⟨9⟩	助詞	
⟨1⟩	→	⟨2⟩	助動詞		⟨9⟩	→	⟨10⟩	名詞	
⟨2⟩	→	⟨3⟩	⟨4⟩		⟨10⟩	→	⟨11⟩	助詞	
⟨3⟩	→	名詞	助詞		⟨11⟩	→	⟨12⟩	名詞	
⟨4⟩	→	⟨5⟩	⟨14⟩		⟨12⟩	→	接頭語	名詞	
⟨5⟩	→	⟨6⟩	助詞		⟨13⟩	→	動詞	語尾	
⟨6⟩	→	⟨7⟩	名詞		⟨14⟩	→	動詞	語尾	助動詞
⟨7⟩	→	⟨8⟩	⟨13⟩						

図 5.4: 構文構造から得られる書き換え規則

⟨12⟩ → 接頭語 名詞

という規則について考える．[接頭語 名詞] という句の主辞は句の一番最後にある “名詞” であると考えられる．そこで，この主辞 “名詞” に “句” をつけたラベル “名詞句” を左辺のノード ⟨12⟩ に与える．同様に，

X → 形容詞句 名詞句

という規則が存在すると仮定し，ラベルの決定されていないノード X に非終端記号を与える場合を考える．この時，[形容詞句 名詞句] という句全体の主辞もまた句の最後にある “名詞句” であると考えられる．先ほどと異なるのは主辞となる記号が非終端記号であるという点である．このような場合には，右再帰を用いて左辺ノード X にも主辞と同じ “名詞句” というラベルを与える．以上のような操作を繰り返し行うことにより，構文構造のラベルに対して非終端記号を自動的に与えることができる．

しかしながら，このようなラベルの与え方が常に適切であるわけではない．

- 主辞にならない品詞

  例えば，

  X → 接続詞 記号

  という規則について考える[3]．[接続詞 記号] という句の 1 番最後にある品詞は “記号” であるが，この句の主辞は “記号” ではなく “接続詞” である．したがって，左辺のノード X に与えるラベルも “記号句” ではなく “接続詞句” とすべきである．このように，“記号” は主辞にはならない品詞であるとみなし，句の一番最後にある要素が “記号” である場合には，その左隣にある要素を主辞とみなす．

- “語尾” と “助動詞” の取り扱い

  図 5.4 の中の

  ⟨13⟩ → 動詞 語尾

  という規則について考える．今までのやり方では，[動詞 語尾] という句の 1 番最後にある品詞は “語尾” であるので，左辺のノード ⟨13⟩ に与えるラベルは “語尾句” となる．ところが，5.2.1 項で述べたように，EDR コーパスにおいては，“語尾” という品詞は動詞の語尾にだけではなく形容詞・形容動詞・助動詞の語尾にも割り当てられている．したがって，このようなラベルの付け方では，

[3] この規則の右辺は「しかし,」などに対応している．

X　→　形容詞　語尾
X　→　形容動詞　語尾
X　→　助動詞　語尾

といった規則の左辺にも “語尾句” というラベルを与えることになる．この場合，“語尾句” というラベルを割り当てられたノードが “動詞”，“形容詞”，“形容動詞”，“助動詞” のどれを含んでいるのかを識別することができない．同様に，規則の右辺の一番最後にある要素が “助動詞” のときも，左辺に “助動詞句” というラベルを与えるのは好ましいことではない．このような理由から，句の一番最後にある要素が品詞 “語尾” または “助動詞” である場合には，その左隣にある要素から左辺に与える非終端記号を導出する．

- 主辞が “助詞” の場合

  左辺に “助詞句” というラベルを与えることも考えられるが，わかりやすさのため “後置詞句” というラベルを与える．

- 主辞が “接尾語” の場合

  EDR コーパスにおいては，品詞が “接尾語” となる形態素は「月」,「日」,「メートル」など単位を表しているものが多く，他にも「区」,「氏」など全体として名詞句を形成するものがほとんどである．そこで，主辞が “接尾語” のときには左辺ノードに “名詞句” というラベルを与える．

以上のようないくつかの例外処理が必要ではあるが，基本的には句の一番最後にある要素を主辞とみなして，それから左辺ノードに与えるラベルを決定する．

本節で提案した括弧付きコーパスから文法を獲得するアルゴリズムを以下にまとめる．

【文法獲得アルゴリズム】

1. 構文構造の中で，まだラベルが割り当てられていなくて，かつその子ノードには全てラベルが割り当てられているノードを見つける．そのようなノードがなければ Step 3. へ．
2. Step 1. で見つけたノードが構文構造のルートである場合には，そのノードのラベルを開始記号 S とする．それ以外は【ラベル決定アルゴリズム】(後述) を用いてノードに与えるラベルを決定する．Step 1. へ戻る．
3. この段階では，構文構造の全ての内部ノードにラベルが与えられている．この構文構造の枝分れを

   ノード　→　子ノードの列

   という形に分解し CFG 規則とする．

【ラベル決定アルゴリズム】

“記号”,“語尾”,“助動詞” 以外の要素で子ノードの列の最も右側にあるものを選び，それを X とする．

- X が “助詞” の場合，左辺ノードに “後置詞句” というラベルを与える．
- X が “接尾語” の場合，左辺ノードに “名詞句” というラベルを与える．
- X が “助詞”，“接尾語” 以外の品詞の場合，左辺ノードに “X 句” というラベルを与える．例えば X が “名詞” の場合，“名詞句” というラベルを与える．
- X が非終端記号の場合，左辺ノードにも同じ X というラベルを与える．例えば X が “名詞句” の場合，左辺ノードにも同じ “名詞句” というラベルを与える．

図 5.3 の内部ノードに対してラベルを与えたのが図 5.5 である．また，これから得られる文法規則を図 5.6 に示す．以上の操作をコーパスの全ての構文構造に対して行うことにより CFG を獲得することができる．

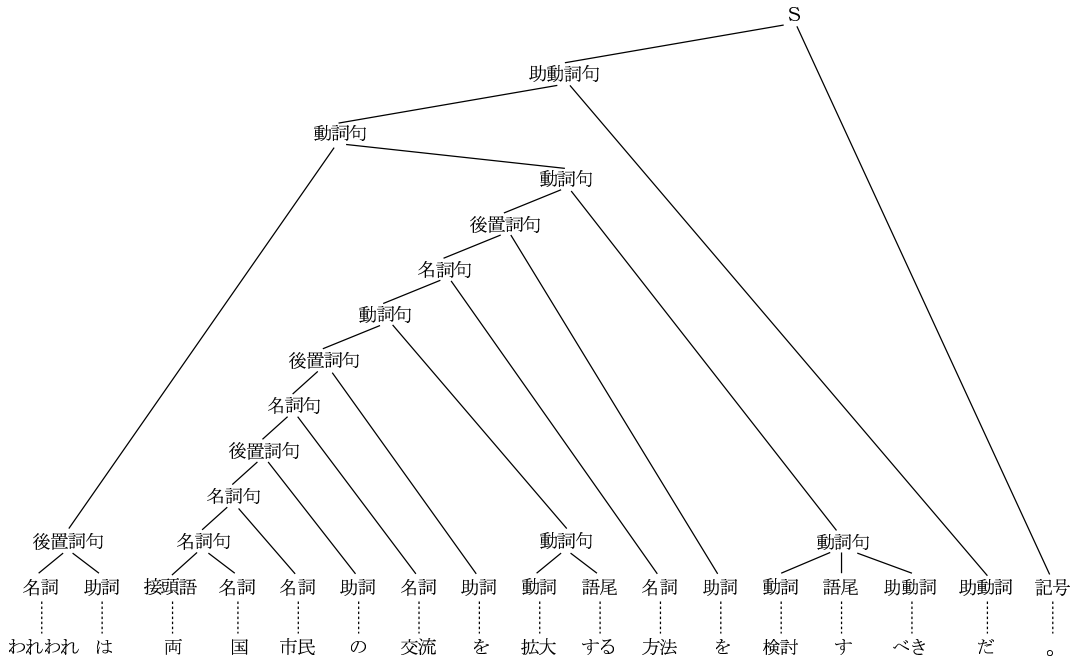


図 5.5: 内部ノードにラベルが与えられた構文構造

次に，本手法の文法獲得に要する計算量について考察する．【文法獲得アルゴリズム】は，「句の主辞はその句における一番最後の要素である」という日本語の言語学的特徴を利用して，括弧付けによる構文構造の内部ノードに非終端記号を決定的に与えているため，文法獲得に必要な計算量はコーパスの構文構造の内部ノード数に比例する．また，長さ n の文があったとき，それに対する最も内部ノード数の多い構文構造は完全な二分木であり，そのときの内部ノード数は $n-1$ である．したがって，文法獲得に必要な計算量は入力文の長さ n にも比例する．

5.3 文法の改良

サイズの小さなコーパスを用いて，前節で説明した方法により CFG を獲得する予備実験を行ったところ，以下のような問題点が明らかになった．

- 得られた文法規則の正当性

  獲得した文法規則を調べてみたところ，一見して誤りと思われる規則がいくつか含まれていることがわかった．

- 文法のサイズが大きい

  EDR コーパスからランダムに選び出した 3,000 例文から CFG を獲得したところ，文法規則の数は 1,009 となり，コーパスサイズに比べて非常に多くの文法規則が抽出されることがわかった．構文解析に要するコストを考えると，文法サイズが不必要に大きいことは望ましいことではない．

S	→	助動詞句	記号	
助動詞句	→	動詞句	助動詞	
動詞句	→	後置詞句	動詞句	
動詞句	→	動詞	語尾	
動詞句	→	動詞	語尾	助動詞
後置詞句	→	名詞	助詞	
後置詞句	→	名詞句	助詞	
名詞句	→	後置詞句	名詞	
名詞句	→	接頭語	名詞	
名詞句	→	動詞句	名詞	
名詞句	→	名詞句	名詞	

図 5.6: 獲得された文法規則

- 生成される解析木の数が多い

  EDR コーパスから，CFG を獲得した 3,000 例文とは異なる 100 例文をランダムに選び出し，獲得した CFG を用いてこれらを構文解析したところ，解析結果の候補として生成された解析木の数は平均 1.5×10^6 となり，非常に多くの解析木を生成することがわかった．また，メモリ不足によって解析に失敗した文は 69 文あった．複数の解析結果の候補 (この場合は解析木) の中から正しい解を選び出す曖昧性解消に要するコストを考えると，文法が生成する解析木の数はできるだけ少ないことが望ましい．

本節ではこれらの問題への対応策について述べる．

5.3.1 不正な規則の検出・削除

獲得した文法規則を調べたところ，以下の 2 つの規則のように一見して明らかに誤りと思われる規則がいくつか見つかった．

後置詞句	→	語尾	助詞
名詞句	→	接尾語	名詞

このような不正な規則は文法から削除すべきである．しかしながら，獲得された文法規則を 1 つずつ調べて不正な規則を見つけ出すのは，それに要する人的負担を考えれば現実的ではない．何らかの方法を用いて不正な規則を自動的に検出することが望ましい．

上記の 2 つの規則は本来句の後ろにあるべき “語尾” や “接尾語” が規則の右辺の先頭に現れているために不正であると判断できる．したがって，“語尾” や “接尾語” など句の末尾にあるべき品詞が右辺の先頭に現れる規則を検出し，それらについてのみ人間が正しいかどうかを判断して不正な規則を文法から削除する方法が考えられる．同様に，句の先頭にあるべき品詞が右辺の末尾に現れる規則についても調べる必要がある．ここで問題となるのは，句の先頭または末尾に現れやすい品詞をどのように特定するかということである．これは人間が判断しても良いのだが，本研究ではコーパスにおける品詞の出現位置の傾向から判断することにした．以下，その方法について説明する．

1. EDR コーパスの括弧付けの中で，品詞のみから構成されているものを取り出す．すなわち，図 5.4 において右辺が品詞のみで構成されている規則を取り出す．

2. 各品詞について以下の R_f，R_l を求める．

$$R_f = \frac{\text{その品詞が 1.の規則の右辺の先頭に現われる回数}}{\text{その品詞が 1.の規則の右辺に現われる回数}} \tag{5.1}$$

$$R_l = \frac{\text{その品詞が 1.の規則の右辺の末尾に現われる回数}}{\text{その品詞が 1.の規則の右辺に現われる回数}} \tag{5.2}$$

R_f の値が大きい品詞は右辺の先頭に頻繁に現われる傾向があり，R_l の値が大きい品詞は右辺の末尾に頻繁に現われる傾向があるとみなすことができる．

3. 本研究では，右辺の先頭に現れやすい品詞を F タイプの品詞，右辺の末尾に現れやすい品詞を L タイプの品詞と呼ぶ．F タイプおよび L タイプの品詞を R_f, R_l の値をもとに特定する．

EDR コーパスの全ての例文の括弧付けによる構文構造から各品詞の R_f，R_l の値を調べた．結果を表 5.1 に示す．

表 5.1: 各品詞の R_f，R_l の値

品詞	R_f	R_l	タイプ
連体詞	0.995865	0.004037	F
接頭語	0.987264	0.010339	F
接続詞	0.978495	0.003515	F
形容動詞	0.974203	0.024439	F
感動詞	0.955882	0.044118	F
副詞	0.947773	0.040692	F
数字	0.929910	0.069401	F
形容詞	0.882726	0.001184	F
動詞	0.791572	0.010428	
名詞	0.722553	0.267825	
助動詞	0.009624	0.614986	L
接尾語	0.001435	0.994015	L
助詞	0.000374	0.770126	L
記号	0.000265	0.694060	L
語尾	0.000100	0.585626	L

本研究では，R_f の値が 0.85 以上の品詞を F タイプの品詞，R_l の値が 0.50 以上の品詞を L タイプの品詞とした．表 5.1 において，F または L のついている品詞がそれぞれ F タイプ，L タイプの品詞である．

次に，この結果をもとに文法の中から不正な規則を除去する方法について説明する．

1. EDR コーパスの括弧付けの中で品詞のみから構成されているもの，すなわち，図 5.4 の中で右辺が品詞のみで構成されている規則について，以下の条件を満たすものを取り出した．

右辺の品詞列の末尾が F タイプの品詞である規則
右辺の品詞列の先頭が L タイプの品詞である規則

その結果，165 種類の規則が検出された．

2. これらの規則の全てが誤りであるわけではない．例えば，

X　→　助動詞　語尾　　(X は構文構造内のノード)

という品詞列は，L タイプの品詞 “助動詞” が右辺の先頭に現れているにも関わらず誤りであるとは認められない．そこで，各品詞列に対応したコーパス中の例文を見てその品詞列が誤りであるかどうかを人手で調べ，32 個の品詞列のみを誤りと判断した．

3. EDR コーパスの例文の中で，2. で見つけた不正な規則を含む例文を取り出したところ，147 文が検出された．これらの例文は括弧付けによる構文構造そのものが間違っていると判断しコーパスから削除した．

この操作により，F タイプの品詞が右辺の末尾に，また L タイプの品詞が右辺の先頭に現れるような不正な規則が獲得されることはない．

5.3.2　文法サイズの縮小

ここでは，コーパスから獲得した文法のサイズを縮小する方法を提案する．文法サイズを縮小する方法としてまず考えられるのは，出現頻度の低い規則を削除することである．しかし，単純に出現頻度の低い規則を削除した場合，その規則がコーパスの構文構造作成時の誤りによって生じた不適切な規則であればよいが，稀にしか現われない言語現象に対応した規則である場合には，そのような規則を削除することにより文法の適用範囲 (coverage) が狭くなる．両者を出現頻度のみで区別することは難しく，出現頻度が低いからといってその規則を削除することは必ずしも適切ではない．

予備実験で獲得した文法を調べたところ，右辺長の長い規則が多く含まれていることがわかった．予備実験で獲得した文法規則の右辺長の分布を表 5.2 に示す．

表 5.2: 文法規則の右辺長の分布

右辺長	2	3	4	5	6	7	8	9	10	11	12	13	14	16
規則数	235	205	155	161	111	69	31	25	7	6	1	1	1	1

右辺長の長い規則が多く含まれていることがわかる．そのような規則の一例を次に挙げる．

動詞句　→　動詞　語尾　名詞　助詞　形容動詞　語尾　動詞　語尾

これは，コーパスのある例文において，

[　動詞　語尾　名詞　助詞　形容動詞　語尾　動詞　語尾　]

といった括弧付けがなされているためである．本来，その例文の構文構造を反映させるためにはもう少し細かい括弧付けが必要である．しかし，EDR コーパスの中には多くの要素を 1 つの括弧で括ってしまう例文も存在する．このような右辺の長い規則の存在が文法サイズを大きくする原因の 1 つとして考えられる．

右辺長の長い規則の場合，文法中の他の規則を用いることによって，その右辺の記号列を生成できる場合がある．例えば，文法中に次のような規則があったとする．

r_b :　動詞句　→　動詞句　後置詞句　動詞句
r_{c1} :　動詞句　→　動詞　語尾
r_{c2} :　後置詞句　→　名詞　助詞
r_{c3} :　動詞句　→　形容動詞　語尾　動詞　語尾

これら 4 つの規則を用いれば，非終端記号 “動詞句” から “動詞 語尾 名詞 助詞 形容動詞 語尾 動詞 語尾” という記号列を生成することが可能である．このことを図式化したものを図 5.7 に示す．このように，ある規則を文法から除去しても，他の規則によってその右辺の記号列を生成できるような場合には，文法の生成能力は変わらない．

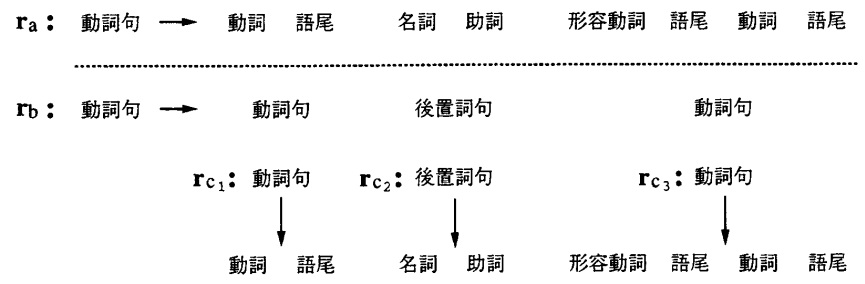


図 5.7: 複数の規則を用いた記号列の展開

そこで,「冗長な規則」を次のように定義する．

> ある規則 $r_i : A_i \rightarrow \zeta_i$ があるとき，文法内の r_i 以外の規則を用いて非終端記号 A_i を記号列 ζ_i に展開できるならば，すなわち $A_i \stackrel{*}{\rightarrow} \zeta_i$ であるならば，r_i は冗長な規則である．

$\stackrel{*}{\rightarrow}$ は規則を 1 回以上適用することを示す．冗長な規則を削除する前の文法によって受理される文は，冗長な規則を削除した後の文法でも必ず受理される．したがって，冗長な規則を自動的に検出しそれを削除すれば，文法の適用範囲を狭めることなく文法サイズを縮小することができる．

本項で提案した冗長な規則を検出しそれを削除するアルゴリズムを以下にまとめる．

【冗長規則削除アルゴリズム】

R, R_{new} を次のように定義する．

R	$\cdots$	獲得した文法規則の集合
R_{new}	$\cdots$	冗長な規則を削除して作られる新しい文法規則の集合 (R の中から冗長でない規則を取り出した集合)

1. R_{new} を空集合とする．
2. R の中から右辺長の一番長い規則 r_a を 1 つ取り出す．
3. 以下の条件を満たす規則の組 $\{r_b^j, r_{c1}^j, \cdots, r_{cn}^j\}$ を可能な限り見つける．

   規則 r_b^j の右辺に含まれる非終端記号 B_i^j を，B_i^j を左辺とする規則 r_{ci}^j の右辺の記号列 β_i^j に置き換えた記号列が r_a の右辺の記号列と一致する．

   この条件を図示すると図 5.8 のようになる．但し，図 5.8 において，A, $B_i \in N$， α_i, $\beta_i \in (N+T)*$ である．(N は非終端記号の集合，T は終端記号の集合)

   ※ このような規則の組が 1 つも見つからなかった場合 ($j = 0$ の場合)

   r_a は冗長な規則ではないので，これを R_{new} に加える．

   ※ このような規則の組が 1 つ以上見つかった場合 ($j >= 1$ の場合)

   r_a は冗長な規則である．このときは r_a を R_{new} には加えず，次のステップへ進む．

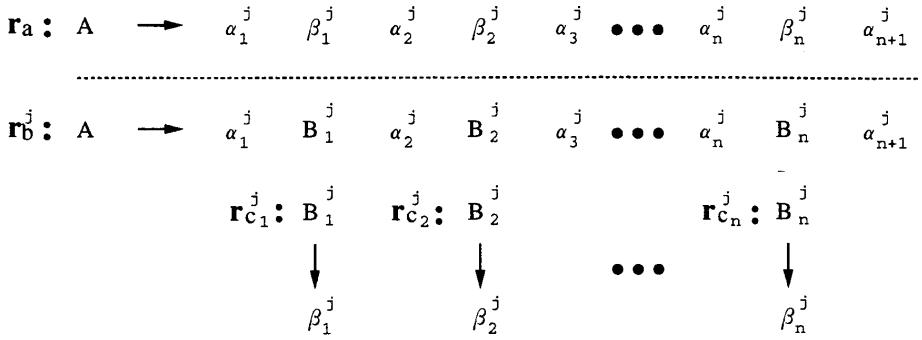


図 5.8: 冗長な規則のチェック

4. R が空なら終了．それ以外は Step 2. へ戻る．

以上のように冗長な規則を削除することにより，文法の適用範囲を狭めることなく文法サイズを縮小することができる．この方法により文法サイズをどの程度縮小することができるのかについては 5.4 節の実験で評価する．

5.3.3 文法が生成する解析木数の抑制

ここでは，獲得した文法が生成する解析木の数を抑制するための 3 つの方法を提案する．

5.3.3.1 同一品詞列の取り扱い

構文解析を行う文の中に同じ品詞が複数並んだ句が存在する場合には，生成される解析木数が増大すると予想される．例えば，“名詞” が 3 つ並んで構成される句の構造としては，名詞間の修飾関係に応じて図 5.9 に示す 3 つの構造が考えられる．

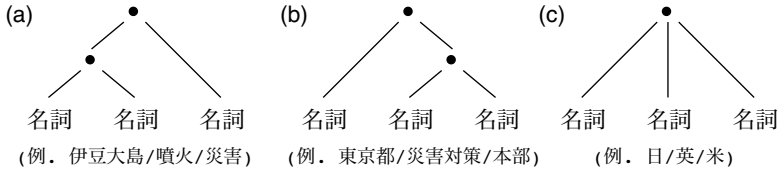


図 5.9: “名詞” が 3 つ並んだ句の構造

ところが，これらの構造の中から正しいものを選択するためには何らかの意味的な情報が必要である [43]．したがって，意味的な情報を全く用いずに構文解析を行う場合は，これらの構造全てを解析結果の候補として生成する．一般に，生成される解析木の数は組合せ的に増大するため，同一品詞列に対して不必要な構造を無意味に生成することが解析木数を増大させる原因の 1 つとなっている．そこで構文解析の段階では，図 5.9 のような構造を全て生成する代わりに，図 5.10 のような右下がりの構造のみを出力することにし，この部分の係り受け解析については構文解析の後で行われる意味解析に任せることにした[4]．また，他

[4] ここでは構文解析と意味解析を独立に行うことを仮定している．本来なら，1.1 節で述べたように，これらは同時に行うことが

の非終端記号と区別するために，図 5.10 の構造の内部ノードには “X 列”(例えば X が “名詞” の場合は “名詞列”) というラベルを与えることにした．このように同一品詞列に対する構造を一意に決めれば解析結果として得られる解析木の数を減少させることができる．

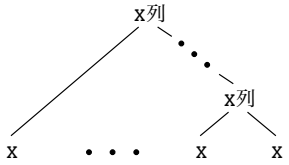


図 5.10: 右下がりの構造

同一品詞列に対して図 5.10 のような右下がりの構造のみを生成するために，5.2.2 項に述べた【文法獲得アルゴリズム】に，次の手続きを最初のステップとして追加する．

【文法獲得アルゴリズム】

0. 構文構造において一種類の品詞のみを支配するノードがあれば，そのノードの下の構造を図 5.10 のような右下がりの構造に修正する．

1. ～ 3.　　変更なし．

また,【ラベル決定アルゴリズム】に次の手続きを追加する．

【ラベル決定アルゴリズム】

- 子ノードが品詞 “X” または非終端記号 “X 列” のみによって構成されている場合には，“X 列” というラベルを与える．

5.3.3.2　品詞の細分化

5.2.1 項で述べたように EDR コーパスで使われている品詞は 15 種類である．したがって，コーパスから獲得した文法に含まれる終端記号 (品詞セット) の数も 15 であるが，これは構文解析を行うのに十分であるとは言えない．例えば，コーパスの中に

[　名詞　助詞　名詞　]　　(e.g.　記者席 / と / 傍聴席　)

という括弧付けが存在し，名詞並列を表わす CFG 規則が獲得されたとする．ところが，この規則は “名詞 助詞 名詞” という品詞列に常に適用され,「地上 / に / 茅 (を出す)」といった名詞並列でない入力に対しても，それが名詞並列であるといった解析結果を出力する．これは全ての助詞に対して同じ “助詞” という品詞を与えているためであり，並列助詞と他の助詞に異なる品詞を与えれば，このような誤った解析を回避することができる．

そこで，EDR コーパスに用いられている品詞を細分化して生成される解析木の数を抑制することを試みた．ここでは “記号” と “助詞” の 2 つの品詞に着目する．

理想的である．しかしながら，本章では CFG の自動獲得を目的としている．したがって，獲得された CFG を統合的確率言語モデルで取り扱うような統合処理に適用するよりも，CFG が主に取り扱う構文解析のみを行った方が，獲得された CFG 規則そのものの品質を評価しやすい．このような理由から，5.4 節の評価実験においても，構文解析と意味解析を独立に行うことを仮定している．

- 品詞 “記号” の細分化

  EDR コーパスにおいては，記号には全て “記号” という品詞が割り当てられている．しかし，読点は文の切れ目を，句点は文の終りを表す特別な記号であり，他の記号とは区別するべきである．そこで，形態素「、」と「,」には “読点” という品詞を与えることにした．また，EDR コーパス中の例文の文末に現れる形態素のほとんどは「。」,「.」,「？」,「！」のいずれかであり，しかもこれらは文末以外に現れることはほとんどなかった．そこで，形態素「。」,「.」,「？」,「！」には “文末記号” という品詞を与えることにした．また，これらの以外の形態素が文末に現れる文，及びこれらの形態素が文末以外の場所に現れる文，合計 102 文を例外としてコーパスから除去した．

- 品詞 “助詞” の細分化

  EDR コーパスにおいては，助詞には全て “助詞” という品詞が割り当てられているが，その助詞の持っている機能により “格助詞”, “並列助詞” などの品詞を割り当てるべきである．しかしながら，助詞の中には 2 つ以上の機能を持つものもあり，助詞の機能をその表層だけから判断することは一般に困難である．そこで，EDR コーパスにおいて “助詞” という品詞を割り当てられた形態素「M」については，その形態素毎に独自の品詞 “助詞M” を割り当てることにした．例えば，形態素「は」が “助詞” という品詞を割り当てられていたならば，その品詞を “助詞は” に変更する．

CFG の獲得は，まずコーパスの品詞を上記のように細分化し，その後で 5.2.2 項で提案した【文法獲得アルゴリズム】 に従って行う．また，品詞の細分化に伴い 5.2.2 項の【ラベル決定アルゴリズム】 を以下のように変更する．下線を引いた部分が変更箇所である．

【ラベル決定アルゴリズム】
<u>“記号”，“語尾”，“助動詞”，“読点”，“文末記号”</u> 以外の要素で子ノードの列の最も右側にあるものを選び，それを X とする．

- X が <u>“助詞M”</u> の場合，左辺ノードに “後置詞句” というラベルを与える．

  (以下同じ)

5.3.3.3 法・様相を表わす助動詞に対する構造の統一

文末に現われる助動詞は文全体の法や様相を表していることが多い．例えば，EDR コーパス中の 2 つの例文

(a) １０月中旬には、袋から顔を出しそうだ。
(b) そのうえソ連は対越援助を削減しそうだ。

には「そう」と「だ」という 2 つの助動詞が含まれている．これらは文全体にそれぞれ伝聞，断定の意味合いを持たせる働きをしている．ところが EDR コーパスにおいては，このような助動詞は，文全体に付加している構造 (図 5.11 の (a)) と，文末の最後の要素に付加している構造 (図 5.11 の (b)) の 2 通りの構造で表されている．このような 2 種類の構文構造を含むコーパスから獲得された文法は，文末に助動詞を含む文に対して少なくとも図 5.11 のような 2 つの構造を生成し，このことが解析木の数を増加させる一因となっている．

そこで，助動詞が文全体に付加された図 5.11 の (a) のような構造を，図 5.11 の (b) のような構造に修正してから文法を獲得することにした．助動詞に対する構造を統一することにより，生成される解析木数

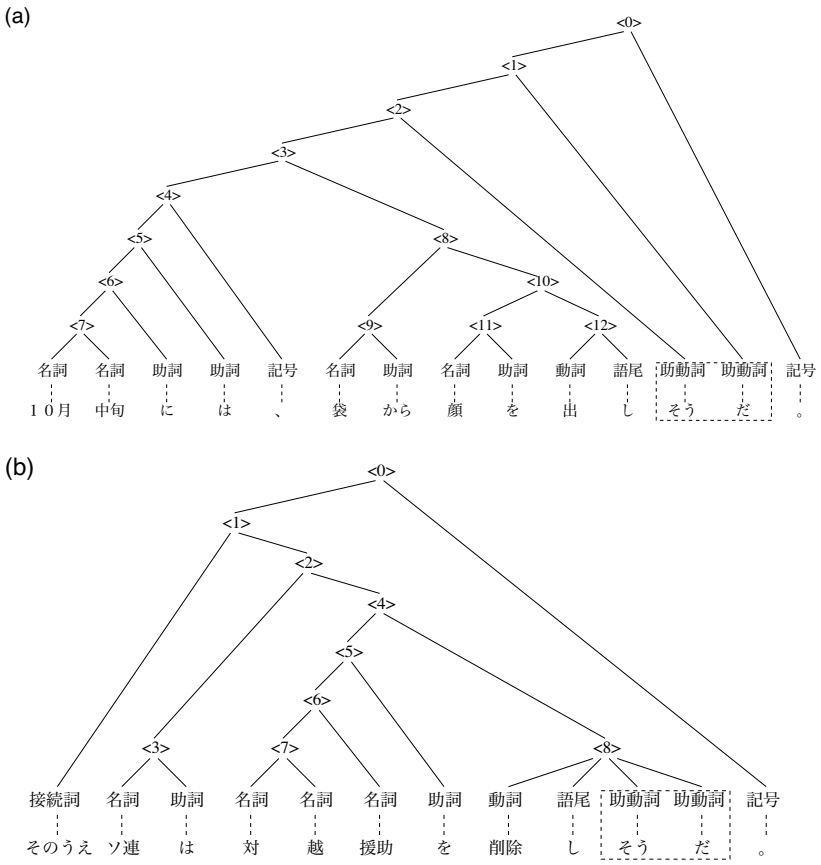


図 5.11: 助動詞に対する 2 つの構造

を減らすことができる．統一後の構造として図 5.11 の (a) ではなく (b) を選択したのは，(a) のような構造からは解析木数を著しく増加させる文法規則が獲得されるからである．例えば，図 5.11 の (a) のノード $\langle 1\rangle$,$\langle 2\rangle$,$\langle 3\rangle$,$\langle 8\rangle$,$\langle 10\rangle$,$\langle 12\rangle$ には，5.2.2 項の【文法獲得アルゴリズム】に従って “動詞句” という非終端記号が割り当てられ，その結果次のような規則が獲得される．

動詞句　→　動詞句　助動詞　　(“$\langle 1\rangle \rightarrow \langle 2\rangle$ 助動詞” という枝分かれに対応)

ところが，この規則により “助動詞” がノード $\langle 8\rangle$,$\langle 10\rangle$,$\langle 12\rangle$ に付加される構造も生成される．これに対して，図 5.11 の (b) のような構造からは上述のような文法規則は獲得されないため，無駄な解析木を生成することはない．

5.4　評価実験

本研究で提案した文法獲得手法の評価実験を行った．まず，EDR コーパスの例文のうち，約 10 分の 1 に相当する 20,000 例文をランダムに選んでテストデータとし，残りを訓練データとした．そして，訓練データから CFG を獲得し，獲得した CFG を用いてテストデータの例文を構文解析することにより，獲得した CFG の品質を評価した．この際，3 章で述べたような統合的確率言語モデルが対象とする統合処理，すな

わち構文解析を形態素解析や意味解析と同時に行うのではなく，構文解析のみを行った．これは，CFG が主に取り扱う構文解析のみを行った方が獲得された文法そのものの品質を評価しやすいためである．また，本研究で提案した統合的確率言語モデルにおいては，構文構造を生成する際にはどのような文法を用いてもよいので，本章で提案した手法により獲得された CFG を統合的確率言語モデルに組み込むことも可能である．

5.4.1 文法獲得

文法獲得を以下の手順で行った．

1. 訓練データの例文の品詞を細分化した．(5.3.3.2)
   また，文末の助動詞に対する構造を統一した．(5.3.3.3)
2. 【文法獲得アルゴリズム】に従って訓練データから CFG を獲得した．(5.2.2 項，5.3.3.1)
3. 【冗長規則削除アルゴリズム】に従って冗長な規則を削除した．(5.3.2 項)

コーパスから獲得した CFG の概要を表 5.3 に示す．

表 5.3: 獲得した CFG

	非終端記号数	終端記号数	規則数
G_0	41	149	15206
G_1	41	149	2219

G_1 は上述の手続きによって訓練データから獲得された CFG，G_0 は冗長規則を削除する前の CFG である．冗長規則を削除したことにより文法サイズを約 85%縮小することができた．

また，各 CFG 規則の確率 $P(r_i)$ を式 (5.3) のように推定することにより，PCFG モデルを学習した [106].

$$r_i : A \rightarrow \zeta_i \text{のとき} \qquad P(r_i) = \frac{O(r_i)}{\sum_{\forall r_j \,:\, A \rightarrow \zeta_j} O(r_j)} \tag{5.3}$$

式 (5.3) において，$O(r_i)$ は規則 r_i の訓練コーパスにおける出現頻度を表わす．すなわち，規則 r_i の出現頻度を A を左辺とする全ての規則の出現頻度の和で割った値を，その規則の確率 $P(r_i)$ とする．

5.4.2 構文解析

獲得された CFG を用いてテスト文の構文解析を行った．構文解析は GLR 法により行った．結果を表 5.4 に示す．

表 5.4: 構文解析結果

受理	不受理	メモリ不足	合計
12,658 文	23 文	7,319 文	20,000 文

“受理” は構文解析に用いた GLR パーザが解析に成功して 1 個以上の解析木を出力したことを，“不受理” は解析に失敗したことを，“メモリ不足” はメモリ不足のために GLR パーザが解析を中断したことを示す．全体の約 36%に当たる 7,319 文がメモリ不足のために解析できなかった．そこで，これらの文については，PCFG モデルで与えられる生成確率の低い部分木を解析途中で破棄する枝刈りを行いながら再度構文解析を行った．その結果を表 5.5 に示す．

表 5.5: 枝刈りを行う構文解析結果

受理	不受理	メモリ不足	合計
5,822 文	562 文	935 文	7,319 文

これにより，$12,658 + 5,822 = 18,480$ 文を受理することができた．受理した文の平均単語数は 24.45 単語であった．また，生成した解析木数の 1 文当たりの平均は 3.24×10^9 であった．非常に多くの解析木が生成されているが，PCFG モデルにより解析木の生成確率を計算し，その上位何位かを出力することによって解析結果の候補数を絞り込むことが可能である．

まず，文法の適用範囲の広さを示す尺度として，受理率を次のように定義する．

$$\text{受理率} = \frac{\text{受理した文の数}}{\text{構文解析した文の数}} \tag{5.4}$$

受理率は 18480/20000 $\simeq$ 0.924 となり，適用範囲の広い文法が得られたことがわかる．また，受理しなかった 1520 文のうち 935 文 (約 61.5%) がメモリ不足によるものである．したがって，GLR パーザの使用メモリを増やすことができれば受理率はさらに向上することが予想される．

次に，GLR パーザが出力した解析木の評価を行った．出力された解析木がどれだけ正しいかを評価するための尺度として，括弧付けの再現率，括弧付けの適合率，文の正解率をそれぞれ以下のように定義した．

$$\text{括弧付けの再現率} = \frac{\text{正しい括弧付けの数}}{\text{コーパスの構文構造に含まれる括弧付けの数}} \tag{5.5}$$

$$\text{括弧付けの適合率} = \frac{\text{矛盾しない括弧付けの数}}{\text{解析木に含まれる全ての括弧付けの数}} \tag{5.6}$$

$$\text{文の正解率} = \frac{\text{出力した解析木の中に正しい解析木が含まれる文の数}}{\text{受理した文の数}} \tag{5.7}$$

ここで “正しい括弧付け” とは，コーパスに付加された構文構造の括弧付けと完全に一致している解析木中の括弧付けを表し，“矛盾しない括弧付け” とは，コーパスに付加された構文構造の全ての括弧付けと交差していない括弧付けを表す [68]．また “正しい解析木” とは，解析木中の全ての括弧付けが矛盾していない解析木を表す．解析木の評価方法としては，コーパスの各例文に付加された構文構造を正解とみなし，これと同じ構造を持つ解析木を正しい解析結果とする方法も考えられる．しかしながら，5.3.2 項で述べたように，EDR コーパスの括弧付けの中には多くの要素を 1 つの括弧で括っているものも含まれる．これに対し，冗長な規則，すなわち右辺長の長い規則を削除した CFG は，EDR コーパスに付加された括弧付けよりも細かく括弧付けする傾向を持っている．したがって，コーパスの構文構造と単純に比較して正しい解析結果か否かを判断するのは適切であるとは言えない．“括弧付けの適合率” 及び “文の正解率” を計算する際に，コーパスに付加された構文構造と完全に一致していなくても，“矛盾しない括弧付け” 及び “矛盾する括弧付けを含まない解析木” を正解としたのはこのためである．

まず，PCFG モデルによって与えられる生成確率が 1 位の解析木について，括弧付けの再現率，括弧付けの適合率，文の正解率を調べた．結果を表 5.6 に示す．

表 5.6: 解析結果の評価 (1 位のみ)

括弧付けの再現率	括弧付けの適合率	文の正解率
54.30%	65.74%	8.47%

Schabes らは，英語の括弧付きコーパス (Wall Street Journal Corpus) から Inside-Outside アルゴリズムにより獲得した文法を用いて構文解析を行い，20〜30 単語のテスト文に対して括弧付けの適合率が 71.5%，文の正解率が 6.8%であったと報告している[5][84]．表 5.6 の実験結果は，括弧付けの適合率では Schabes らの結果に劣るが文の正解率では優っている．しかしながら，本研究とは使用しているコーパスや対象言語が異なるため，単純な比較はできない．

次に，生成確率の上位 k 位の解析木を出力し，その中から矛盾する括弧付けの最も少ない解析木を選んで評価した．結果を表 5.7 に示す．

表 5.7: 構文解析結果の評価 (上位 k 位)

k	括弧付けの再現率	括弧付けの適合率	文の正解率
1	54.30%	65.74%	8.47%
5	57.60%	69.04%	16.23%
10	59.53%	71.10%	21.11%
20	61.46%	73.18%	26.51%
30	62.48%	74.13%	29.06%

上位 30 位までの解析木を出力した場合，その中に正解となる解析木が含まれている文の割合は 8.47%から 29.06%に向上することがわかった．

最後に，構文解析を行う文法のサイズを変化させ，受理率と正解率および生成される解析木数との相関を調べる実験を行った．まず，G_1 の中からある一定の閾値 P 以下の確率を持つ文法規則を除去し，サイズの小さい文法 G_2〜G_5 を獲得した．また，閾値以下の規則を削除した後，残された規則の出現回数をもとに各規則の確率の推定をやり直した．次に，テストデータの中から枝刈りなしで受理した 12,658 文 (表 5.4 参照) を G_2〜G_5 を用いて構文解析し，結果を比較した．テスト文をこのように限定したのは，パーザがメモリ不足によって構文解析を中断した場合には文法が生成する解析木の数を測定することができないからである．解析した文の平均単語数は 19.01 単語であった．

実験結果を表 5.8 に示す．“平均解析木数” は PCFG が生成する 1 文当りの解析木の数であり，いわゆる統語的曖昧性を表わす．メモリ不足によって解析に失敗した文はなかった．括弧付けの再現率，括弧付けの適合率，文の正解率は，生成確率の 1 位の解析木のみについて評価した．表 5.8 により，文法サイズが小さくなるにつれて受理率が低下していることがわかる．また，受理率の低下に伴い平均解析木数も減少する傾向が見られる．これに対し，受理率が変化しても括弧付けの再現率，適合率，文の正解率はほとんど変化していない．このことから，受理率を向上させるために文法サイズを大きくして，その結果得られる解析木の数が増大しても，生成確率の上位の解析木のみを出力すれば正解率はほとんど変わらないといえる．

[5] 彼らは括弧付けの再現率は示していない．

表 5.8: 文法サイズと解析結果の変化

文法 (閾値 P)	G_1 (—)	G_2 (10^{-5})	G_3 (10^{-4})	G_4 (10^{-3})	G_5 (10^{-2})
非終端記号数	41	37	34	23	15
終端記号数	111	80	57	33	23
規則数	2,219	1,289	871	390	115
受理率	100%	99.39%	95.57%	76.74%	34.22%
平均解析木数	2.730×10^7	1.810×10^7	1.071×10^7	9.841×10^5	4.485×10^4
括弧付けの再現率	62.71%	62.73%	62.66%	62.91%	60.11%
括弧付けの適合率	75.59%	75.62%	75.68%	76.58%	73.64%
文の正解率	12.07%	12.00%	12.25%	13.38%	11.45%

第6章　結論

本論文では，まず形態素解析，構文解析，語義曖昧性解消を統合的に処理する際に生じる曖昧性を解消するために，複数の統計情報を同時に取り扱う統合的確率言語モデルを提案した．この統合的確率言語モデルの特徴は，構文的優先度，単語の出現頻度，語彙的従属関係などといった曖昧性解消に有効な様々な種類の統計情報を独立に取り扱う点にある．これにより，それぞれの統計情報を別々の言語資源から学習できるだけでなく，学習した個々の統計情報が曖昧性解消にどのように働くかを容易に理解することができる．また，この統合的確率言語モデルの評価実験として，構文モデル及び語彙モデルを異なる言語資源から個別に学習し，日本語文の文節の係り受け解析実験を行った．この結果，構文的な統計情報と語彙的な統計情報のそれぞれが曖昧性解消に大きく貢献することを確認した．さらに，係り受け解析に失敗した事例を調査した結果，今回の実験においては考慮されていなかったが曖昧性解消に有効であると考えられる語彙的従属関係を新たに確率モデルに組み込むことや，入力文が並列構造を含む場合や補助用言を含む場合などを十分考慮に入れる必要があることが明らかになった．また，これらの問題についても，本研究で提案した統合的確率言語モデルによって十分対処することが可能である．

次に，自然言語処理に適した特性を持つと考えられる最大エントロピー法を統計情報の学習に適用することを目的に，このアルゴリズムを効率化する手法を提案した．具体的には，確率モデルの推定を行う GIS アルゴリズムの効率化と，確率モデルの推定に有効な素性を選択する素性選択アルゴリズムの効率化を行った．さらに，有効な素性を選択する新たな手法として，素性効用に基づく素性選択アルゴリズムを提案した．最大エントロピー法によって助詞の生成確率を学習する実験を行ったところ，素性効用に基づいて素性を選択した場合は，素性選択アルゴリズムを用いた場合に比べて，はるかに少ない計算量で確率モデルを推定することができ，また推定された確率モデルの品質もほぼ等しいことがわかった．

一方，様々な言語現象に対応した自然言語処理用知識を言語資源から獲得する研究として，文脈自由文法を括弧付きコーパスから自動獲得する手法を提案した．本研究で提案した手法は，日本語の主辞が句の一番最後の要素であるという言語学的な知識を利用することにより，文法獲得に要する計算量を抑制したことに特徴がある．また，コーパスから抽出した文法を洗練するいくつかの手法も提案した．本研究で提案した方法によって抽出・洗練された文法を用いた構文解析実験の結果，適用範囲の広い文法を獲得できたことがわかった．

最後に，本研究の今後の課題について述べる．まず，本研究で提案した統合的確率言語モデルの特性をさらに調査することが挙げられる．特に，3.2 節で行った統合的確率言語モデルの評価実験は，文節の係り受け解析のみを対象としており，またテスト文の文長も比較的短い．したがって，形態素解析と構文解析を同時に行ったり，より文長の長い入力文を対象にするなど，大規模な実験を行ったときに，統合的確率言語モデルが曖昧性解消にどの程度有効であるのか実験的に評価しなければならない．また，3.2.4 項では，統合的確率言語モデルを利用した曖昧性解消における問題点とその対処方法について述べたが，これらを基に統合的確率言語モデルをさらに洗練し，曖昧性解消の精度を向上させることも課題のひとつとして挙げられる．一方，最大エントロピー法の効率化に関しては，確率モデルの推定に用いる素性集合 F を選択する手法を更に洗練する必要がある．本研究では，素性が確率モデルの推定にどの程度有効であるかを評価する尺度として素性効用を提案した．しかしながら，この素性効用に基づく素性選択は，他の素性が素性集合 F に含まれているか否かを考慮していない点に問題がある．このような素性同士の依存関係を考慮し，か

つ実際の自然言語処理に適用できる程度の計算量で素性選択を行う手法を探究していく必要がある．また，本論文では，括弧付きコーパスから文法を効率良く自動獲得する手法を提案したものの，獲得した文法を統合的確率言語モデルに組み込むことは行っていない．コーパスから自動獲得した文法を統合的確率言語モデルに組み込むことによって，獲得された文法の総合的な評価を行うことも今後の課題である．

本研究のような統計的自然言語処理に関する研究を進めていく上で注意しなければならないのは，純粋に言語資源のみから統計情報を学習したり自然言語処理用知識を獲得することには限界があるということである．すなわち，コーパスなどの実データだけでなく，既に広く知られている言語学的な知見やヒューリスティクスも積極的に利用することによって，より品質の高い統計情報や自然言語処理用知識を学習することができる．本研究においても，統合的確率言語モデルにおいて単語生成文脈を決定する際や，文法の自動獲得において木構造の内部ノードのラベルを決定する際に，このような言語学的知識を利用している．人間が持っている知識と実データから獲得できる情報をうまく融合して自然言語処理における様々な問題の解決を図ることが，統計的自然言語処理における重要な指針のひとつとして挙げられる．

謝辞

本研究を進めるにあたり，多くの方々の御指導・御支援を賜りました．

まず，本研究を終始親切に御指導下さいました指導教官の徳永健伸助教授に心より感謝いたします．また，私との討論に多くの時間を割いていただき，数多くの貴重な御意見をいただきました乾健太郎助手に深く感謝いたします．

審査教官である古井貞熙教授，沼尾正行助教授，丸山宏助教授には本研究に対し貴重な助言をいただきました．心から感謝いたします．

NHK 放送技術研究所の江原暉将氏には，最大エントロピー法について色々と教えを賜りました．改めて御礼を申し上げます．

NTT コミュニケーション科学研究所知識処理研究部翻訳処理研究グループの皆様には，実験で使用した日本語語彙体系を提供していただきました．また，京都大学助手の黒橋禎夫氏には，京大コーパスを正式リリース前に提供していただきました．ここに深く感謝いたします．

田中・徳永研究室の皆様には討論会などを通して，折に触れ貴重な御意見をいただきました．秋葉友良氏 (現，電子技術総合研究所)，小林義行氏 (現，日立製作所中央研究所)，秋山典丈氏 (現，株式会社キーエンス)，上條俊一氏 (現，NTT データ通信株式会社)，藤井敦氏には，自然言語処理全般に関して様々な助言をいただきました．田中淳志氏 (現，東京大学大学院)，遠藤貴将氏には共同研究者として御協力いただきました．Timothy John Baldwin 氏には英語論文の校正で大変お世話になりました．Virach Sornlertlamvanich 氏，橋本泰一氏には実験で使用した MSLR パーザを提供して下さいました．熊野正氏 (現，NHK 放送技術研究所)，杉山聡氏 (現，NTT 基礎研究所)，今井宏樹氏，植木正裕氏，鈴木泰山氏には計算機環境の面で色々と相談に乗っていただきました．田中・徳永研究室の歴代の秘書の方々にも大変お世話になりました．

最後になりましたが，田中穂積教授には常日頃から研究に関する貴重な御意見や温かい励ましをいただき，公私にわたりまして大変お世話になりました．改めて厚く御礼申し上げます．

参考文献

[1] 相沢道雄, 徳永健伸, 田中穂積. 一般化 LR 法を用いた形態素解析と統語解析の統合. 電子情報通信学会技術報告 (NLC93–2), pp. 9–16, 1993.

[2] Eduardo de Paiva Alves. The selection of the most probable dependency structure in Japanese using mutual information. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 372–374, 1996.

[3] Timothy Baldwin, Takenobu Tokunaga, and Hozumi Tanaka. Semantic verb classes in the analysis of head gapping in Japanese relative clauses. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pp. 409–414, 1997.

[4] Timothy Baldwin, Takenobu Tokunaga, and Hozumi Tanaka. Syntactic and semantic constraints on head gapping in Japanese. 言語処理学会第 3 回年次大会発表論文集, pp. 209–212, 1997.

[5] Adam L. Berger, Stephen A. Della Pietra, and Vincent. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, Vol. 22, No. 1, pp. 39–71, 1996.

[6] Ezra Black, Fred Jelinek, John Lafferty, David M. Magerman, Rercer Mercer, and Salim Roukos. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 31–37, 1993.

[7] Rens Bod. A computational model of language performance: Data oriented parsing (DOP). In *Proceedings of the 14th International Conference on Computational Linguistics*, Vol. 3, pp. 855–859, 1992.

[8] Rens Bod. Monte calro parsing. In *Proceedings of the International Workshop on Parsing Technologies*, pp. 1–11, 1993.

[9] Rens Bod. Using an annotated corpus as a stochastic grammar. In *Proceedings of the 6th Conference of European Chapter of the Association for Computational Linguistics*, pp. 37–44, 1993.

[10] Rens Bod. Using an annotated language corpus as a virtual stochastic grammar. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pp. 778–783, 1993.

[11] Eric Brill. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 259–265, 1993.

[12] Ted Briscoe and John Carroll. Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, Vol. 19, No. 1, pp. 25–59, 1993.

[13] Eugene Charniak, Curtis Hendrickson, Neil Jacobson, and Mike Perkowitz. Equations for part-of-speech tagging. In *Proceedings of the National Conference on Artificial Intelligence*, pp. 784–789, 1993.

[14] Eugene Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the National Conference on Artificial Intelligence*, 1997.

[15] Mahesh Chitrao. *Statistical Technique for Parsing Message*. PhD thesis, New York University, 1990.

[16] Michael Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 1996.

[17] Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 16–23, 1997.

[18] Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun. A practical part-of-speech tagger. In *Proceedings of the Conference on Applied Natural Language Processing*, pp. 133–140, 1992.

[19] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The annals of Mathematical Statistics*, Vol. 43, No. 5, pp. 1470–1480, 1972.

[20] 江原暉将. 最大エントロピー法を用いて n グラム確率をバイグラム確率で補完する方法. 言語処理学会第 2 回年次大会発表論文集, pp. 369–372, 1996.

[21] 江原暉将. 最大エントロピー法を用いてバイグラム確率から n グラム確率を求める. 情報処理学会自然言語処理研究会, Vol. 96, No. 56, pp. 25–30, 1996.

[22] Jason M. Eisner. An empirical comparison of probability models for dependency grammar. Technical Report IRCS-96-11, Institute for Research in Cognitive Science, Univ. of Pennsylvania, 1996.

[23] Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the International Conference on Computational Linguistics*, pp. 340–345, 1996.

[24] 藤尾正和, 松本裕治. EDR 括弧付きコーパスを利用した、統計的日本語係り受け解析. EDR 電子化辞書利用シンポジウム論文集, pp. 49–55, 1997.

[25] 藤尾正和, 松本裕治. 統計的手法を用いた係り受け解析. 情報処理学会自然言語処理研究会, Vol. 97, No. 4, pp. 83–90, 1997.

[26] Donald Hindle and Mats Rooth. Structural ambiguity and lexical relations. *Computational Linguistics*, Vol. 19, No. 1, pp. 103–120, 1993.

[27] 平岡冠二, 松本裕治. コーパスからの動詞の格フレーム獲得と名詞のクラスタリング. 情報処理学会自然言語処理研究会, Vol. 94, No. 98, pp. 79–86, 1994.

[28] Wide Roeland Hogenout and Yuji Matsumoto. Experiments with using semantical categories in parsing systems. 言語処理学会第 2 回年次大会発表論文集, pp. 381–384, 1996.

[29] 本田岳夫, 奥村学. コーパスから二重確率的シソーラスを自動的に獲得する手法の提案. 情報処理学会自然言語処理研究会, Vol. 95, No. 89, pp. 33–38, 1995.

[30] 池原悟, 宮崎正弘, 横尾昭男. 日英機械翻訳のための意味解析用の知識とその分解能. 情報処理学会論文誌, Vol. 34, No. 8, pp. 1692–1704, 1993.

[31] 池原悟, 宮崎正弘, 白井諭, 横尾昭男, 中岩浩己, 小倉健太郎, 大山芳史, 林良彦. 日本語語彙体系 — 全5巻 —. 岩波書店, 1997.

[32] 乾健太郎, 白井清昭, 徳永健伸, 田中穂積. 種々の制約を統合した統計的日本語文解析. 情報処理学会自然言語処理研究会, Vol. 96, No. 114, pp. 35–42, 1996.

[33] Kentaro Inui, Kiyoaki Shirai, Hozumi Tanaka, and Takenobu Tokunaga. Integrated probabilistic language modeling for statistical parsing. Technical Report TR97-0005, Dept. of Computer Science, Tokyo Institute of Technology, 1997.

[34] Kentaro Inui, Kiyoaki Shirai, Takenobu Tokunaga, and Hozumi Tanaka. Integration of statistical techniques for parsing. In *summary collection of the IJCAI'97 poster session*, 1997.

[35] Kentaro Inui, Virach Sornlertlamvanich, Hozumi Tanaka, and Takenobu Tokunaga. A new formalization of probabilistic GLR parsing. In *Proceedings of the International Workshop on Parsing Technologies*, 1997.

[36] Kentaro Inui, Virach Sornlertlamvanich, Hozumi Tanaka, and Takenobu Tokunaga. A new probabilistic LR language model for statistical parsing. Technical Report TR97-0004, Dept. of Computer Science, Tokyo Institute of Technology, 1997.

[37] F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi, and S. Roukos. Decision tree parsing using a hidden derivation model. In *Proceedings of the Workshop: Human Language Technology*, pp. 272–277, 1994.

[38] K. Kita, T. Morimoto, K. Ohkura, S. Sagayama, and Y. Yano. Spoken sentence recognition based on HMM-LR with hybrid language modeling. *IEICE Trans. Inf. & Syst.*, Vol. E77–D, No. 2, 1994.

[39] Masaki Kiyono and Jun-ichi Tsujii. Linguistic knowledge acquisition from parsing failures. In *Proceedings of the 6th Conference of European Chapter of the Association for Computational Linguistics*, pp. 222–231, 1993.

[40] Masaki Kiyono and Jun'ichi Tsujii. Combination of symbolic and statistical approaches for grammatical knowledge acquisition. In *Proceedings of the Conference on Applied Natural Language Processing*, pp. 72–77, 1994.

[41] Masaki Kiyono and Jun'ichi Tsujii. Hypothesis selection in grammar acquisition. In *Proceedings of the 14th International Conference on Computational Linguistics*, Vol. 2, pp. 837–841, 1994.

[42] Yosiyuki Kobayasi, Takenobu Tokunaga, and Hozumi Tanaka. Analysis of Japanese compound nouns using collocational information. In *Proceedings of the 14th International Conference on Computational Linguistics*, Vol. 2, pp. 865–869, 1994.

[43] 小林義行, 徳永健伸, 田中穂積. 名詞間の意味的共起情報を用いた複合名詞の解析. 自然言語処理, Vol. 3, No. 1, pp. 29–43, 1996.

[44] Julian Kupiec. Robust part-of-speech tagging using a hidden markov model. *Computer Speech and Language*, Vol. 6, No. 3, pp. 225–242, 1992.

[45] Julian Kupiec and John Maxwell. Training stochastic grammars from unlabelled text corpora. In *Workshop on Statistically-Based Natural Language Programming Techniques*, pp. 8–13, 1992.

[46] 黒橋禎夫, 長尾眞. 並列構造の検出に基づく長い日本語文の構文解析. 自然言語処理, Vol. 1, No. 1, pp. 35–57, 1994.

[47] 黒橋禎夫, 長尾眞. 京都大学テキストコーパス・プロジェクト. 人工知能学会全国大会論文集, pp. 58–61, 1997.

[48] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer speech and languages*, Vol. 4, pp. 35–56, 1990.

[49] Hang Li and Naoki Abe. Generalizing case frames using a thesaurus and the MDL priciple. In *Proceedings of the Recent Advances in Natural Language Processing*, 1995.

[50] 李航. 心理言語学原理に基づいた確率的曖昧性解消法. コンピュータソフトウェア, Vol. 13, No. 6, pp. 489–501, 1996.

[51] Hang Li. A probabilistic disambiguation method based on psycholinguistic principles. In *Proceedings of the Workshop on Very Large Corpora*, pp. 141–154, 1996.

[52] Hang Li and Naoki Abe. Clustering words with the MDL priciple. In *Proceedings of the International Conference on Computational Linguistics*, pp. 4–9, 1996.

[53] Hang Li and Naoki Abe. Learning dependencies between case frame slots. In *Proceedings of the International Conference on Computational Linguistics*, pp. 10–15, 1996.

[54] Hang Li and Naoki Abe. Learning dependencies between case frame slots. 情報処理学会自然言語処理研究会, Vol. 96, No. 114, pp. 93–99, 1996.

[55] David M. Magerman and Mitchell P. Marcus. Pearl: A probabilistic chart parser. In *Proceedings of the Conference of European Chapter of the Association for Computational Linguistics*, pp. 15–20, 1991.

[56] David M. Magerman. *Natural Language Parsing as a Statistical Pattern Recognition.* PhD thesis, Stanford University, 1994.

[57] David M. Magerman. Statistical decision-tree models for parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 276–283, 1995.

[58] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, Vol. 19, No. 2, pp. 313–330, 1993.

[59] H. Maruyama and S. Ogino. A statistical property of Japanese phrase-to-phrase modification. *Mathematical Linguistics*, pp. 348–352, 1992.

[60] 松本裕治, 黒橋禎夫, 宇津呂武仁, 妙木裕, 長尾眞. 日本語形態素解析システム JUMAN 使用説明書 version 2.0. Technical report, 京都大学工学部長尾研究室，奈良先端科学技術大学院大学 松本研究室, 1994.

[61] 三原健一. 日本語の統語構造. 松柏社, 1994.

[62] 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. 情報処理学会自然言語処理研究会, Vol. 110, pp. 79–86, 1995.

[63] 村上仁一, 嵯峨山茂樹. HMM を用いた形態素解析. 情報処理学会第 45 回全国大会講演論文集, pp. 161–162, 1992.

[64] 長尾真 編. 自然言語処理. 岩波講座 ソフトウェア科学 15. 岩波書店, 1996.

[65] 永田昌明. 前向き DP 後向き A*アルゴリズムを用いた確率的日本語形態素解析システム. 情報処理学会自然言語処理研究会, Vol. 94, No. 47, pp. 73–80, 1994.

[66] Masaaki Nagata. A stochastic Japanese morphological analyzer using a forward-DP backward-A* N-Best search algorithm. In *Proceedings of the International Conference on Computational Linguistics*, pp. 201–207, 1994.

[67] 日本電子化辞書研究所. EDR 電子化辞書仕様説明書第２版. Technical Report TR–045, 1995.

[68] Fernando Pereira and Yves Schabes. Inside-Outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 128–135, 1992.

[69] L. R. Rabiner and B. H. Juang. An introduction to hidden markov models. In *Proceedings of the 1986 International Conference on Acoustics, Speech, and Signal Processing*, pp. 4–16, 1986.

[70] Adwait Ratnaparkhi, Salim Roukos, and R. Todd Ward. A maximum entropy model for parsing. In *Proceedings of the International Conferrence on Spoeken Language Processing*, pp. 803–806, 1994.

[71] Adwait Ratnaparkhi, Jeff Reyner, and Salim Roukos. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the Workshop: Human Language Technology*, pp. 250–255, 1994.

[72] Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of the the Emprical Methods in Natural Language Processing Conference*, 1996.

[73] Adwait Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the the Emprical Methods in Natural Language Processing Conference*, 1997.

[74] Adwait Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing. Technical Report IRCS Report 1997–08, University of Pennsylvania, 1997.

[75] Real World Computing Partnership. RWC text database. `http://www.rwcp.or.jp/wswg.html`, 1995.

[76] Philip Resnik. Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In *Proceedings of the 14th International Conference on Computational Linguistics*, Vol. 2, pp. 418–424, 1992.

[77] Philip Resnik. Semantic classes and syntactic ambiguity. In *Proceedings of the Workshop: Human Language Technology*, pp. 278–283, 1993.

[78] Jeffrey C. Reynar and Adwaint Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Conference on Applied Natural Language Processing*, 1997.

[79] Yasubumi Sakakibara. An efficient learning of context-free grammars for bottom-up parsers. In *Proceedings of the International Conference On Fifth Generation Computer Systems*, pp. 447–454, 1988.

[80] Yasubumi Sakakibara. Learning context-free grammars from structural data in polynominal time. In *Proceedings of the 1988 Workshop on Computational Learning Theory*, pp. 330–344, 1988.

[81] 佐藤健吾, 中西正和. 最大エントロピー法による対訳単語対の抽出. 情報処理学会情報処理学会自然言語処理研究会, Vol. 97, No. 109, pp. 21–27, 1997.

[82] Yves Schabes. Stochastic lexicalized tree-adjoining grammars. In *Proceedings of the 14th International Conference on Computational Linguistics*, Vol. 2, pp. 425–432, 1992.

[83] Yves Schabes, Anne Abeille, and Aravind K. Joshi. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *Proceedings of the 12th International Conference on Computational Linguistics*, Vol. 2, pp. 578–583, 1988.

[84] Yves Schabes, Michael Roth, and Randy Osborne. Parsing the wall street journal with the Inside-Outside algorithm. In *Proceedings of the 6th Conference of European Chapter of the Association for Computational Linguistics*, pp. 341–347, 1993.

[85] Yves Schabes and Richard C. Waters. Lexicalized context-free grammars. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 121–129, 1993.

[86] Satoshi Sekine and Ralph Grishman. A corpus-based probabilistic grammar with only two non-terminals. In *Proceedings of the International Workshop on Parsing Technologies*, 1995.

[87] 島津明, 内藤昭三, 野村浩郷. 助詞「の」が結ぶ名詞の意味関係の subcategorization. 情報処理学会自然言語処理研究会, Vol. 86, No. 6, pp. 1–8, 1986.

[88] 白井清昭, 徳永健伸, 田中穂積. コーパスからの文法の自動抽出. 情報処理学会自然言語処理研究会, Vol. 94, No. 47, pp. 81–88, 1994.

[89] Kiyoaki Sirai, Takenobu Tokunaga, and Tanaka Hozumi. Automatic extraction of Japanese grammar from a bracketed corpus. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pp. 211–216, 1995.

[90] 白井清昭, 徳永健伸, 田中穂積. EDR コーパスからの確率文脈自由文法の自動抽出に関する研究. EDR 電子化辞書利用シンポジウム論文集, pp. 57–62, 1995.

[91] 白井清昭, 徳永健伸, 田中穂積. 構文構造付きコーパスからの確率文脈自由文法の自動抽出. 情報処理学会第 50 回全国大会講演論文集, pp. 61–62, 1995.

[92] 白井清昭, 乾健太郎, 徳永健伸, 田中穂積. 最大エントロピー法を用いた単語 bigram の推定. 情報処理学会自然言語処理研究会, Vol. 96, No. 114, pp. 21–28, 1996.

[93] 白井清昭, 乾健太郎, 徳永健伸, 田中穂積. 統計的日本語文解析における種々の統計量の扱いについて. 自然言語処理シンポジウム「大規模資源と自然言語処理」, 1996. http://www.etl.go.jp/etl/nl/nlsympo/96/.

[94] Kiyoaki Shirai, Kentaro Inui, Tanaka Hozumi, and Takenobu Tokunaga. An empirical study on statistical disambiguation of Japanese dependency structures using a lexically sensitive language model. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pp. 215–220, 1997.

[95] 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, pp. 125–146, 1997.

[96] 白井清昭, 乾健太郎, 徳永健伸, 田中穂積. 最大エントロピー法による格の従属関係の学習. 言語処理学会第 3 回年次大会発表論文集, pp. 337–340, 1997.

[97] 白井清昭, 乾健太郎, 徳永健伸, 田中穂積. 統合的確率モデルを用いた日本語文解析. 情報処理学会自然言語処理研究会, Vol. 97, No. 109, pp. 43–50, 1997.

[98] Virach Sornlertlamvanich, Kentaro Inui, Kiyoaki Shirai, Hozumi Tanaka, and Takenobu Tokunaga. Empirical evaluation of probabilistic GLR parsing. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pp. 169–174, 1997.

[99] Virach Sornlertlamvanich, Kentaro Inui, Kiyoaki Shirai, Hozumi Tanaka, Takenobu Tokunaga, and Toshiyuki Takezawa. Incorporating probabilistic parsing into an LR parser – LR table engineering (4) –. 情報処理学会情報処理学会自然言語処理研究会, Vol. 97, No. 53, pp. 61–68, 1997.

[100] Virach Sornlertlamvanich, 乾健太郎, 田中穂積, 徳永健伸. 確率つき一般化 LR 構文解析について. 言語処理学会第 3 回年次大会発表論文集, pp. 301–304, 1997.

[101] 田辺利文, 冨浦洋一, 日高達. 語の共起関係の文脈自由文法への取り込み法. EDR 電子化辞書利用シンポジウム論文集, pp. 25–31, 1995.

[102] 田中淳志. 形態素タグつきコーパスからの単語共起情報の獲得. 卒業論文, 東京工業大学 工学部 情報工学科, 1996.

[103] Masaru Tomita. *Generalized LR Parsing.* Kluwer Academic, 1991.

[104] 宇津呂武仁, 宮田高志, 松本裕治. 最大エントロピー法による下位範疇化の確率モデル学習および統語的曖昧性解消による評価. 情報処理学会自然言語処理研究会, Vol. 97, No. 53, pp. 69–76, 1997.

[105] K. Vijay-Shanker and Aravind K. Joshi. Some computational properties of tree adjoining grammars. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pp. 82–93, 1985.

[106] C. S. Wetherell. Probabilistic languages: A review and some open questions. *Computing surveys*, Vol. 12, No. 4, pp. 361–379, 1980.

[107] 横田和章, 安部賢司, 青島直哉, 藤崎博也. コーパスに基づく日本語文法の自動獲得. 言語処理学会第2回年次大会発表論文集, pp. 169–172, 1996.

[108] 横田和章, 亀田弘之, 藤崎博也. 日本語の文法および未知の認知単位の自動獲得のための一方法. 自然言語処理, Vol. 3, No. 4, pp. 115–128, 1996.

[109] 国立国語研究所. 分類語彙表, 増補版, 1996.

付 録 A　目標事象を二分したときのエントロピーの計算

式 (4.45) の導出を以下に示す.

$$
\begin{aligned}
H_{(T|h)} &\stackrel{def}{=} -\sum_{t\in T} P(t|h)\log P(t|h) && \text{(A.1)}\\
&= -\sum_{t\in T_f} P(t|h)\log P(t|h) - \sum_{t\in \overline{T_f}} P(t|h)\log P(t|h) && \text{(A.2)}\\
&= -\sum_{t\in T_f} P(T_f|h)P(t|T_f,h)\log P(T_f|h)P(t|T_f,h)\\
&\quad -\sum_{t\in \overline{T_f}} P(\overline{T_f}|h)P(t|\overline{T_f},h)\log P(\overline{T_f}|h)P(t|\overline{T_f},h) && \text{(A.3)}\\
&= -P(T_f|h)\sum_{t\in T_f} P(t|T_f,h)\{\log P(T_f|h)+\log P(t|T_f,h)\}\\
&\quad -P(\overline{T_f}|h)\sum_{t\in \overline{T_f}} P(t|\overline{T_f},h)\{\log P(\overline{T_f}|h)+\log P(t|\overline{T_f},h)\} && \text{(A.4)}\\
&= -P(T_f|h)\sum_{t\in T_f} \big\{P(t|T_f,h)\log P(T_f|h)+P(t|T_f,h)\log P(t|T_f,h)\big\}\\
&\quad -P(\overline{T_f}|h)\sum_{t\in \overline{T_f}} \big\{P(t|\overline{T_f},h)\log P(\overline{T_f}|h)+P(t|\overline{T_f},h)\log P(t|\overline{T_f},h)\big\} && \text{(A.5)}\\
&= -P(T_f|h)\Big\{\sum_{t\in T_f} P(t|T_f,h)\log P(T_f|h)+\sum_{t\in T_f} P(t|T_f,h)\log P(t|T_f,h)\Big\}\\
&\quad -P(\overline{T_f}|h)\Big\{\sum_{t\in \overline{T_f}} P(t|\overline{T_f},h)\log P(\overline{T_f}|h)+\sum_{t\in \overline{T_f}} P(t|\overline{T_f},h)\log P(t|\overline{T_f},h)\Big\} && \text{(A.6)}\\
&= -P(T_f|h)\Big\{\sum_{t\in T_f} P(t|T_f,h)\log P(T_f|h)-H_{(T_f|h)}\Big\}\\
&\quad -P(\overline{T_f}|h)\Big\{\sum_{t\in \overline{T_f}} P(t|\overline{T_f},h)\log P(\overline{T_f}|h)-H_{(\overline{T_f}|h)}\Big\} && \text{(A.7)}\\
&= -P(T_f|h)\sum_{t\in T_f} P(t|T_f,h)\log P(T_f|h)+P(T_f|h)H_{(T_f|h)}\\
&\quad -P(\overline{T_f}|h)\sum_{t\in \overline{T_f}} P(t|\overline{T_f},h)\log P(\overline{T_f}|h)+P(\overline{T_f}|h)H_{(\overline{T_f}|h)} && \text{(A.8)}\\
&= -P(T_f|h)\log P(T_f|h)\sum_{t\in T_f} P(t|T_f,h)+P(T_f|h)H_{(T_f|h)}
\end{aligned}
$$

$$
\begin{aligned}
& -P(\overline{T_f}|h)\log P(\overline{T_f}|h)\sum_{t\in\overline{T_f}} P(t|\overline{T_f},h) + P(\overline{T_f}|h)H_{(\overline{T_f}|h)} && \text{(A.9)}\\
= \quad & -P(T_f|h)\log P(T_f|h) + P(T_f|h)H_{(T_f|h)} \\
& -P(\overline{T_f}|h)\log P(\overline{T_f}|h) + P(\overline{T_f}|h)H_{(\overline{T_f}|h)} && \text{(A.10)}\\
= \quad & P(T_f|h)H_{(T_f|h)} + P(\overline{T_f}|h)H_{(\overline{T_f}|h)} + H'_{(T_f,\overline{T_f})} && \text{(A.11)}
\end{aligned}
$$

式 (A.11) は，目標事象の集合 T を T_f と $\overline{T_f}$ とに二分したときの $H_{(T|h)}$ の計算方法を示している．すなわち，ある特定の履歴事象 h についての条件付き確率 $P(t|h)$ のエントロピー $H_{(T|h)}$ は，T_f および $\overline{T_f}$ 内部のエントロピー $H_{(T_f|h)}$，$H_{(\overline{T_f}|h)}$ の重み和と，$T_f, \overline{T_f}$ のどちらに属する目標事象が現われるかを予測する確率分布 $P(X|h)$ $(X \in \{T_f, \overline{T_f}\})$ のエントロピー $H'_{(T_f,\overline{T_f})}$ の和によって計算することができる．