自動構築した大規模格フレームに基づく構文・格解析の統合的確率モデル

河原 大輔[†]　　黒橋 禎夫[††]

本稿では、格フレームに基づき構文・格解析を統合的に行う確率モデルを提案する。格フレームは、ウェブテキスト約 5 億文から自動的に構築した大規模なものを用いる。確率モデルは、述語項構造を基本単位とし、それを生成する確率であり、格フレームによる語彙的な選好を利用するものである。ウェブのテキストを用いて実験を行い、特に述語項構造に関連する係り受けの精度が向上することを確認した。また、語彙的選好がどの程度用いられているかを調査したところ、60.7%という高い割合で使われていることがわかり、カバレージの高さを確認することができた。

キーワード： 格フレーム, コーパス, ウェブ, 構文解析, 格解析

A Fully-Lexicalized Probabilistic Model for Japanese Syntactic and Case Structure Analysis

Daisuke Kawahara[†]　　Sadao Kurohashi[††]

This paper presents an integrated probabilistic model for Japanese syntactic and case structure analysis. Syntactic and case structure are simultaneously analyzed based on wide-coverage case frames that are constructed from a huge raw corpus in an unsupervised manner. This model selects the syntactic and case structure that has the highest generative probability. We evaluate both syntactic structure and case structure. In particular, the experimental results for syntactic analysis on web sentences show that the proposed model significantly outperforms known syntactic analyzers.

KeyWords: *case frame, corpus, web, syntactic analysis, case structure analysis*

1　はじめに

近年、構文解析は高い精度で行うことができるようになった。構文解析手法は、ルールベースのもの (e.g., (黒橋, 長尾 1994))、統計ベースのもの (e.g., (工藤, 松本 2002)) に大別することができるが、どちらの手法も基本的には、形態素の品詞・活用、読点や機能語の情報に基づいて高精度を実現している。例えば、

(1)　弁当を食べて出発した

[†] 情報通信研究機構, National Institute of Information and Communications Technology
[††] 京都大学大学院情報学研究科, Graduate School of Informatics, Kyoto University

という文は、「弁当を → 食べて」のように正しく解析できる。これは、「〜を」はほとんどの場合もっとも近い用言に係るという傾向を考慮しているからである。このような品詞や機能語などの情報に基づく係り受け制約・選好を、ルールベースの手法は人手で記述し、統計ベースの手法はタグ付きコーパスから学習している。しかし、どちらの手法も語彙的な選好に関してはほとんど扱うことができない。

(2) a. 弁当を出発する前に食べた
　　b. 弁当は食べて出発した

(2a) では、「弁当を」が (1) と同じように扱われ、「弁当を → 出発する」のように誤って解析される。(2b) においては、「〜は」が文末など遠くの文節に係りやすいという傾向に影響されて、やはり「弁当は → 出発した」のように誤って解析されてしまう。これらの場合、「弁当を食べる」のような語彙的選好が学習されていれば正しく解析できると思われる。統計的構文解析器においては多くの場合、語彙情報が素性として考慮されているが、それらが用いている数万文程度の学習コーパスからでは、データスパースネスの影響を顕著に受け、語彙的選好をほとんど学習することができない。

さらに、2 項関係の語彙的選好が十分に学習されたとしても、次のような例を解析することは難しい。

(3)　太郎が食べた花子の弁当

「弁当を 食べる」「花子が 食べる」という語彙的選好を両方とも学習しているとすると、「食べた」の係り先はこれらの情報からでは決定することができない。この例文を正しく解析するには、「食べた」は「太郎が」というガ格をもっており、ヲ格の格要素は被連体修飾詞「弁当」であると認識する必要がある。このように、語彙的選好を述語項構造としてきちんと考慮できれば構文解析のさらなる精度向上が期待できる。

述語項構造を明らかにする格解析を実用的に行うためには、語と語の関係を記述した格フレームが不可欠であり、それもカバレージの大きいものが要求される。そのような格フレームとして、大規模ウェブテキストから自動的に構築したものを利用することができる (河原, 黒橋 2006)。本稿では、この大規模格フレームに基づく構文・格解析の統合的確率モデルを提案する。本モデルは、格解析を生成的確率モデルで行い、格解析の確率値の高い構文構造を選択するということを行う。

構文解析手法として、語彙的選好を明示的に扱うものはこれまでにいくつか提案されてきた。白井らと藤尾らは、数十〜数百万文のコーパスから語の共起確率を推定し利用している (白井, 乾, 徳永, 田中 1998; 藤尾, 松本 1999)。本研究にもっとも関連している研究として、阿辺川らによる構文解析手法がある (阿辺川, 奥村 2006)。阿辺川らは、同じ用言を係り先とする格要素間の従属関係と、格要素・用言間の共起関係を利用した構文解析手法を提案している。これら 2 つの関係を新聞記事 30 年分から収集し、PLSI を用いて確率推定を行っている。既存の構文

解析器の出力する n-best の構文木候補に対して、確率モデルに基づくリランキングを適用し、もっとも確率値の高い構文木を選択している。この手法は、PLSI を用いることによって潜在的な意味クラスを導入し、確率を中規模のコーパスから推定している。

本研究は、これらの研究に対して次の点で異なる。

- 明示的に意味、用法が分類された格フレームを用いている。解析時に格フレームを選択することにより、用言の意味的曖昧性を解消し、その意味、用法下において正確な格解析を行うことができる。
- 非常に大規模なコーパスから構築された格フレームを用いることによって、用例の出現を汎化せずに用いている。
- 阿辺川らの手法のように n-best 解をリランキングするのではなく、構文、格構造を生成する生成モデルを定義している。

2　ウェブから獲得した大規模格フレーム

格フレームは、ウェブから収集した大規模コーパスを用いて、(河原, 黒橋 2006) の手法により自動構築を行う。本節では、格フレーム構築手法の概要を述べる。

人間のもつ常識的知識の重要な部分である格フレームは、様々な言語現象をカバーすることが望ましい。そのような格フレームを構築するために、大規模コーパスから漸進的に確からしい情報を抽出する。

まず最初に、大規模コーパスを構文解析し、その解析結果から第 1 段階の格フレームを構築する。格フレームを構築する際の最大の問題は、用言の用法の曖昧性である。つまり、同じ表記の用言でも複数の意味、用法をもち、とりうる格や用例が異なる。例えば、以下の 2 つの例は、用言は「積む」で同じであるが用法が異なっている。

(4) a. トラックに荷物を積む
　　b. 経験を積む

用法が異なる格フレームを別々につくるために、我々は、格フレーム収集の単位を用言とその直前の格要素の組とした。「積む」の例では、「荷物を積む」「経験を積む」を単位として格フレームを収集する。さらに、「荷物を積む」「物資を 積む」などかなり類似している格フレームをマージするためにクラスタリングを行う。

上記の第 1 段階の構築手法では構文解析を用いているために、基本的に格助詞の付属している格要素を収集している。このため、得られる格フレームは、二重主語構文、外の関係、格変化のような複雑な言語現象には対処できないという問題がある。この問題に対処するために、上記で得られた格フレームを用いて再度テキストを解析し、新たな情報を格フレームに与える。新たに得られる情報は、1 回目の格フレーム構築では扱うことができなかった係助詞句 (「〜は」や「〜も」) や被連体修飾詞に関する関係である。

(5)　この車はエンジンが良い

例えば、上例において、構文解析の段階では「車は」は解釈できなかったが、格解析では「{ エンジン } が よい」という格フレームを用いることによって、格フレームにガ格以外の格がないことから「車は」は 2 つ目のガ格であり、「{ エンジン } が よい」は二重主語構文をとることがわかる。

(6)　その問題は彼が図書館で調べている

この例文の「問題は」は、すでに得られている格フレーム「{ 問題, 課題 } を { 図書館 } で調べる」のヲ格の用例群に合致するため、格解析ではヲ格と解析されるだけで、新しい情報は得られない。同様に、被連体修飾詞は構文解析では扱われないが、格解析では、格フレームのガ格、ヲ格などの用例と類似しているかどうか調べることによって解釈される。例えば、「業務を営む免許」の「免許」は、格フレーム「{ 銀行, 会社 } が { 業務, ビジネス } を 営む」のどの格の用例とも類似せず、外の関係と呼ばれる関係をもっていると判定され、この情報が格フレームに加えられる。

上記の手法を用いて、ウェブから収集した約 5 億日本語文から格フレームを構築した。約 350CPU の計算機グリッドを用いてこの処理を行い、約 1 週間で格フレームを構築することができた。この格フレームは約 90,000 用言からなる。その一部を表 1 に示す。

3　構文・格解析の統合的確率モデル

本論文で提案する構文・格解析統合モデルは、入力文がとりうるすべての構文構造に対して確率的格解析を行い、もっとも確率値の高い格解析結果をもつ構文構造を出力する。すなわち、入力文 S が与えられたときの構文構造 T と述語項構造 L の同時確率 $P(T, L|S)$ を最大にするような構文構造 T_{best} と述語項構造 L_{best} を出力する。次のように、$P(S)$ は一定であるので、本モデルは $P(T, L, S)$ を最大にすることを考える。

$$\begin{aligned}(T_{best}, L_{best}) &= \underset{(T,L)}{\operatorname{argmax}} P(T, L|S) \\ &= \underset{(T,L)}{\operatorname{argmax}} \frac{P(T, L, S)}{P(S)} \\ &= \underset{(T,L)}{\operatorname{argmax}} P(T, L, S) \end{aligned} \tag{1}$$

3.1　構文・格解析の統合的確率モデルの概略

本論文では、依存構造に基づく確率的生成モデルを提案する。本モデルは「節」を基本単位とし、主節 (文末の節) から順次生成していく。「節」とは、用言 1 つと、それと関係をもつ格要

表 1: 自動構築した格フレームの例

	格	用例 (数字は頻度を表す)
焼く (1)	ガ	私:18, 人:15, 職人:10, ···
	ヲ	パン:2484, 肉:1521, ケーキ:1283, ···
	デ	オーブン:1630, フライパン:1311, トースター:668, ···
焼く (2)	ガ	先生:3, 政府:3, 人:3, ···
	ヲ	手:2950
	ニ	攻撃:18, 行動:15, 息子:15, ···
焼く (3)	ガ	メーカー:1, ディストリビューター:1, ···
	ヲ	データ:178, ファイル:107, コピー:9, ···
	ニ	R:1583, CD:664, CDR:3, ···
⋮	⋮	⋮
泳ぐ (1)	ガ	イルカ:142, 生:50, 魚:28, ···
	ヲ	海:1188, 水中:281, 海中:101, ···
	デ	クロール:86, 平泳ぎ:49, 泳法:24, ···
⋮	⋮	⋮
磨く (1)	ガ	私:4, 男性:4, 人:4, おれ:4, ···
	ヲ	歯:5959, 奥歯:27, 前歯:12
	デ	ブラシ:38, トイレ:15, 塩:13, ···
⋮	⋮	⋮
録画 (1)	ガ	旦那:4, 妹:2, 知人:2, 友人:2, ···
	ヲ	番組:1435, 放送:521, 特番:26, ···
	ニ	ビデオ:3753, ディスク:256, ···
⋮	⋮	⋮

素群を意味する。$P(T, L, S)$ は、文に含まれる節 c_i を生成する確率の積として次のように定義する。

$$P(T, L, S) = \prod_{c_i \in S} P(c_i | b_h) \tag{2}$$

ここで、b_h は節 c_i の係り先文節である。主節は係り先をもたないが、仮想的な係り先を EOS とする。

従来研究のほとんどは、文生成の確率を、2 文節間の係り受け確率の積としていたが、本研究では式 (2) のように、節、つまり用言と格要素群を単位として生成するモデルとしている。そのため、複数の格要素を考慮して係り受けを決定することができ、例 (3) のような文も正しく

解析できるようなモデルとなっている。

例えば「弁当は食べて目的地に出発した。」という文を考える。「弁当は」が「食べて」に係る場合には、2 つの節「弁当は食べて」「目的地に出発した。」があり、次の確率を考える。

$$P(\text{目的地に出発した。}|\text{EOS}) \times P(\text{弁当は食べて} \mid \text{出発した。})$$

「弁当は」が「出発した。」に係る場合には、2 つの節「食べて」「弁当は目的地に出発した。」があり、次の確率を考える。

$$P(\text{弁当は目的地に出発した。}|\text{EOS}) \times P(\text{食べて} \mid \text{出発した。})$$

本モデルは、これらのうちもっとも確率の高い構造を採用する。

節 c_i は、述語項構造 CS_i と用言タイプ f_i に分解して考える。用言タイプとは、用言の活用や付属語列を意味する。そのため、述語項構造 CS_i に含まれる用言は原型である。係り先の文節 b_h も同様に、語 w_h とタイプ f_h に分けて考える。

$$\begin{aligned} P(c_i|b_h) &= P(CS_i, f_i|w_h, f_h) \\ &= P(CS_i|f_i, w_h, f_h)P(f_i|w_h, f_h) \\ &\approx P(CS_i|f_i, w_h)P(f_i|f_h) \end{aligned} \tag{3}$$

この近似は、用言は係り先文節のタイプには依存しない、また用言タイプは係り先の語には依存しないと考えられるからである。

例えば、$P(\text{弁当は食べて} \mid \text{出発した。})$ は次のようになる。

$$P(CS(\text{弁当は食べる})|\ \text{テ形}, \text{出発する}) \times P(\text{テ形} \mid \text{タ形。})$$

ただし、本モデルにおいて、副詞、連体詞、および連体修飾句は述語項構造に入れず、考慮しない。これらは用言に対して格関係を持たないので、用言格フレームにおいて扱うことができず、生成することができないためである。これらの係り先は、読点がなければ直近の係りうる文節とするなどといったルールに基づいて決定する (黒橋, 長尾 1994)。

式 (3) の $P(CS_i|f_i, w_h)$ を述語項構造生成確率、$P(f_i|f_h)$ を用言タイプ生成確率と呼び、これらについて次の 2 つの節で説明する。

3.2　述語項構造生成確率

述語項構造の生成モデルは、その述語項構造にマッチする格フレームの選択と、入力側の各格要素の格フレームへの対応付けを同時に行うモデルである。

述語項構造 CS_i は、述語 v_i、格フレーム CF_l、格の対応関係 CA_k の 3 つからなると考える。格の対応関係 CA_k とは、図 1 に示すように、入力側の格要素と格フレームの格との対応付

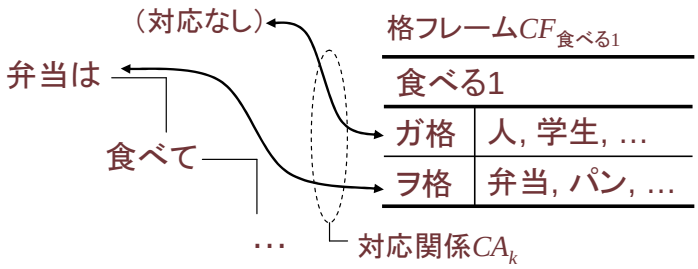

図 1: 格の対応関係 CA_k の例

け全体を表す。対応関係は図示のもの以外にも、「弁当は」をガ格に対応付ける可能性がある。述語項構造生成確率 $P(CS_i|f_i, w_h)$ は次のようになる。

$$\begin{aligned} P(CS_i|f_i, w_h) &= P(v_i, CF_l, CA_k|f_i, w_h) \\ &= P(v_i|f_i, w_h) \\ &\quad \times P(CF_l|f_i, w_h, v_i) \\ &\quad \times P(CA_k|f_i, w_h, v_i, CF_l) \\ &\approx P(v_i|w_h) \qquad \text{(用言生成確率)} \\ &\quad \times P(CF_l|v_i) \qquad \text{(格フレーム生成確率)} \\ &\quad \times P(CA_k|CF_l, f_i) \qquad \text{(格の対応関係生成確率)} \end{aligned} \tag{4}$$

この近似は、述語 v_i はその係り先の語 w_h のみに、格フレーム CF_l は述語 v_i のみに、格の対応関係 CA_k は格フレーム CF_l と付属語列 f_i に依存すると考えられることによる。

用言生成確率と格フレーム生成確率は大規模コーパスの格解析結果から推定する。$P(CA_k|CF_l, f_i)$ は、格の対応関係生成確率と呼び、以下で詳説する。

格の対応関係生成確率

格の対応関係 CA_k を、格フレームの格スロット s_j ごとに考える。格スロット s_j に入力側の格要素 (体言 n_j, 格要素タイプ f_j) が対応付けられているかどうかで場合分けすると、次のように書き換えることができる。

$$\begin{aligned} &P(CA_k|CF_l, f_i) = \\ &\quad \prod_{s_j:A(s_j)=1} P(A(s_j)=1, n_j, f_j|CF_l, f_i, s_j) \\ &\quad \times \prod_{s_j:A(s_j)=0} P(A(s_j)=0|CF_l, f_i, s_j) \end{aligned} \tag{5}$$

ただし、$A(s_j)$ は、格スロット s_j に入力側格要素が対応付けられていれば 1、そうでなければ 0 をとる関数である。

式 (5) 右辺第 1 項の各確率は次のように分解できる。

$$P(A(s_j)=1, n_j, f_j|CF_l, f_i, s_j) = \\ P(A(s_j)=1|CF_l, f_i, s_j) \times P(n_j, f_j|CF_l, f_i, A(s_j)=1, s_j) \quad (6)$$

この式の第 1 項と式 (5) 第 2 項の各確率は、f_i には依存しないと考えられるので、それぞれ $P(A(s_j)=1|CF_l, s_j)$、$P(A(s_j)=0|CF_l, s_j)$ となる。これらは格スロット生成確率と呼び、大規模コーパスの格解析結果から推定する。$P(n_j, f_j|CF_l, f_i, A(s_j)=1, s_j)$ は格要素生成確率と呼ぶ。

例えば、$P(CS(\text{弁当は食べる})|\ \text{テ形}, \text{出発する})$ について考える。「食べる」のある格フレーム $CF_{\text{食べる }1}$ がガ格とヲ格をもっているならば、この格フレームを用いたときの述語項構造生成確率としては、「弁当は」をガ格またはヲ格に対応付けるときの 2 つを考えることになる。以下に「弁当は」をヲ格に対応付けるときの確率を示す。

$$\begin{aligned} &P(CS(\text{弁当は食べる})|\ \text{テ形}, \text{出発する}) = \\ &\quad P(\text{食べる}\ |\ \text{出発する}) \\ &\times P(CF_{\text{食べる }1}|\ \text{食べる}) \\ &\times P(A(\text{を})=1|CF_{\text{食べる }1}, \text{を}) \\ &\times P(A(\text{が})=0|CF_{\text{食べる }1}, \text{が}) \\ &\times P(\text{弁当}, \text{は}\ |CF_{\text{食べる }1}, \text{テ形}, A(\text{を})=1, \text{を}) \end{aligned}$$

格要素生成確率

格要素の体言 n_j と格要素タイプ f_j を生成する確率は独立であり、表層格の解釈は格フレームに依存しないと考え、格要素生成確率は以下のように近似する。

$$P(n_j, f_j|CF_l, f_i, A(s_j)=1, s_j) \approx P(n_j|CF_l, A(s_j)=1, s_j) \times P(f_j|s_j, f_i) \quad (7)$$

$P(n_j|CF_l, A(s_j)=1, s_j)$ は用例生成確率と呼び、格フレーム自体から推定する。

格要素タイプ f_j としては、表層格 c_j、読点の有無 p_j、提題助詞「は」の有無 t_j の 3 つを考慮する。

$$\begin{aligned} P(f_j|s_j, f_i) &= P(c_j, t_j, p_j|s_j, f_i) \\ &= P(c_j|s_j, f_i) \\ &\quad \times P(p_j|s_j, f_i, c_j) \\ &\quad \times P(t_j|s_j, f_i, c_j, p_j) \\ &\approx P(c_j|s_j) \qquad (\text{表層格生成確率}) \\ &\quad \times P(p_j|f_i) \qquad (\text{読点生成確率}) \\ &\quad \times P(t_j|f_i, p_j) \quad (\text{提題助詞生成確率}) \end{aligned} \quad (8)$$

この近似は、c_j は s_j のみに、p_j は f_i のみに、t_j は f_i と p_j に依存すると考えられるためである。表層格生成確率は、表層格を解釈した格をタグ付けした京都テキストコーパス (河原, 黒橋, 橋田 2002) を用いて推定する。

日本語では、読点や提題助詞はそれらの属する文節が遠くに係る場合に用いられやすいという傾向がある。このような傾向を考慮して、読点生成確率 $P(p_j|f_i)$ と提題助詞生成確率 $P(t_j|f_i, p_j)$ を以下のように定義する。

$$P(p_j|f_i) = P(p_j|o_i, u_i) \tag{9}$$
$$P(t_j|f_i, p_j) = P(t_j|o_i, u_i, p_j) \tag{10}$$

o_i は、対象格要素がほかの係り先候補を越えて v_i に係る場合に 1 をとり、それ以外では 0 となる。u_i は、節の区切れとしての強さであり、強い節ほど読点や提題助詞をもつ句を受けやすい。節の強さとしては、南による節の分類 (南 1993) を参考にして設定した 5 段階を考える。

3.3　用言タイプ生成確率

用言タイプ生成確率 $P(f_i|f_h)$ は、文節 b_h のタイプを条件にしたときに、それに係っている節 c_i の用言タイプを生成する確率である。この確率は、節 c_i が連用節であるか連体節であるかで次のように異なる。

節 c_i が連用節の場合は、節間の係り受けに大きな影響を及ぼすと考えられる読点の有無と連用節のタイプ (強さ) を考慮する。これに加えて、c_i がほかの係り先候補を越えて b_h に係るかどうかを考慮する。

$$P_{VBmod}(f_i|f_h) = P_{VBmod}(p_i, u_i|p_h, u_h, o_h) \tag{11}$$

節 c_i が連体節である場合は、受側すなわち体言のタイプには依存しないと考え、次のように定義する。

$$P_{NBmod}(f_i|f_h) = P_{NBmod}(p_i|o_h) \tag{12}$$

4　実験

提案手法によって解析した構文・述語項構造の評価実験を行った。各パラメータは表 2 のリソースから最尤推定によって計算した。これらのリソースは一度の処理で得られたものではなく、構文解析、格フレーム構築、格解析という順番で処理を行い、得られたものである。ここにおける格解析は、シソーラスに基づく類似度を用いた格解析 (河原, 黒橋 2005) である。格フレームはウェブテキスト約 5 億文から自動構築したものを用い、格解析済みデータはウェブテキスト約 600 万文を格解析することによって得たものを用いた。構文構造の候補としては、ルールベースの構文解析器 KNP が出力するすべての候補を用いた。

表 2: 各パラメータの推定

確率		推定に用いるリソース
読点生成確率	$P(p_j\|o_i, u_i)$	京都テキストコーパス
提題助詞生成確率	$P(t_j\|o_i, u_i, p_j)$	京都テキストコーパス
用言タイプ生成確率	$P(p_i, u_i\|p_h, u_h, o_h)$	京都テキストコーパス
表層格生成確率	$P(c_j\|s_j)$	京都テキストコーパス (関係)
用言生成確率	$P(v_i\|w_h)$	構文解析済みデータ
用例生成確率	$P(n_j\|CF_l, A(s_j) = 1, s_j)$	格フレーム
格フレーム生成確率	$P(CF_l\|v_i)$	格解析済みデータ
格スロット生成確率	$P(A(s_j) = \{0, 1\}\|CF_l, s_j)$	格解析済みデータ

4.1 構文解析実験

構文解析実験は、ウェブテキスト 675 文[1]を形態素解析器 JUMAN に通した結果を提案システムに入力することによって行う。その 675 文には、京都テキストコーパスと同じ基準で係り受けのタグ付けを行い、これを用いて係り受けの評価を行った。文末から 2 つ目までの文節以外の係り受けを評価し、その評価結果を表 3 に示す。表において、「CaboCha」とは、SVM に基づく統計的構文解析器 CaboCha[2]を表し、「KNP」とは、構文解析器 KNP を表しており、いずれのシステムにも同じ形態素解析結果を入力している。係り受けの精度比較のため、「CaboCha」には「KNP」による文節区切りの結果を入力し、文節区切りも一致させている。

表 3 より、提案手法は「CaboCha」や「KNP」より精度がよいことがわかる。マクネマー検定を行った結果、提案手法の精度は「CaboCha」と「KNP」より有意 ($p < 0.05$) に上回っていることがわかった。また、表には、係り受けのタイプごとの精度も併せて示してある。述語項構造と密接に関係しているのは、「体言 → 用言」の係り受けであり、その中で中心的なのは「係助詞句以外」である。その精度は「KNP」と比べて 1.6%向上しており、エラー率は 10.9%減少している。これより提案手法が、述語項構造に関係する係り受けの解析に有効であることがわかる。

表 4 に、「KNP」では誤りになるが、提案手法によって正解になった例を挙げる。四角形で囲まれた文節の係り先が×下線部から○下線部に変化したことを示している。

また、以下に提案手法の主な誤り原因を挙げる。

係り受けの正解基準からのずれ

提案モデルは、語彙的な選好を強く考慮して係り受けを決定する。しかし、解析結果が、係

[1] これらの文は格フレーム構築とモデル学習には用いていない。

[2] http://chasen.org/~taku/software/cabocha/ (形態素解析器 JUMAN の結果を入力できる最後のバージョンである CaboCha 0.36 を用いた。)

表 3: 構文解析の精度

	CaboCha	KNP	提案手法
すべて	3,412/3,976 (85.8%)	3,447/3,976 (86.7%)	3,477/3,976 (87.4%)
体言 → 用言	1,308/1,547 (84.6%)	1,310/1,547 (84.7%)	1,328/1,547 (85.8%)
係助詞句	233/298 (78.2%)	244/298 (81.9%)	242/298 (81.2%)
それ以外	1,075/1,249 (86.1%)	1,066/1,249 (85.3%)	1,086/1,249 (86.9%)
体言 → 体言	512/556 (92.1%)	525/556 (94.4%)	526/556 (94.6%)
用言 → 用言	596/760 (78.4%)	593/760 (78.0%)	601/760 (79.1%)
用言 → 体言	452/497 (90.9%)	453/497 (91.1%)	457/497 (92.0%)

表 4: 正解例

- ここは貝の宝庫だそうで夏場は多くの人が [くると] 通りかかった× おばちゃんにききました。○
- 水力発電は、[水]が 高い× ところから低いところへ 流れる○ ときの力を使って、水車を回し、発電機が回って電気を作ります。
- えび篭漁船が余市港に入港した後、すぐに標識用の [エビを] 同港に 停泊した× 当場所属調査船「おやしお丸」に 搬送し、○ 船上で標識の装着作業を行いました。
- 男の空色のはずの [法衣が、] 濃紺に 見える○ ほどに。×
- MOX燃料の中のプルトニウムの量と原子ろに入れるMOX燃料の量を調整する [ことで、] これまでのウラン燃料と同じぐらいの安全性が 確保される○ ことが 確認されています。×

り受けの正解基準とずれるために、誤りとなる場合がある。

- [行政相談委員は、] いつでも自宅でみなさんからのご相談に 応じていますが、× この期間中は次のところで行政相談所を 開きます。○

この文において、「行政相談委員は、」の正解係り先は「開きます。」であるが、提案手法は係り先を「応じていますが、」と解析し、誤りとなる。「開きます。」、「応じていますが、」のどちらも意味的には係り先として正しいと考えられるが、基準としては文末の「開きます。」であるのでずれが生じる。このような問題を解決するには、省略関係の正解を考慮しながら評価を行う必要がある。

係り受けの制約

KNP が出力している構文構造の候補中に、正解の構造が含まれていないことがある。

- 本当に、美味い [コーヒーを] お探しの○ 方にオススメの サイトです。×

表 5: 格解析の精度

	ベースライン	提案手法
係助詞句	72/105 (68.6%)	82/105 (78.1%)
被連体修飾詞	107/155 (69.0%)	121/155 (78.1%)

この文において、「コーヒーを」の正解係り先は「お探しの」であるが、「お探しの」は「コーヒーを」の係り先候補にはなっていないために解析が誤る。「お探しの」のような体言の文節は、通常、連体修飾しか受けないためこのような扱いになっているが、この問題を解決するためにはこのような制約を緩める、より多くの候補を探索する必要がある。

各確率の重み付け

提案モデルにおいて、各確率を重み付けすることは行っていない。実際には、読点を考慮する確率と用例を生成する確率のどちらかを強く考慮するかの重みを最適化した方がよい場合があり、機械学習手法を用いてそのような最適化を行うことが考えられる。

4.2　格解析実験

述語項構造が正しく認識されているかを評価するために、係助詞句と被連体修飾詞の格が正しく認識できているかどうかを調べた。ウェブテキスト 215 文に対して京都コーパスと同様の基準で関係タグを付与し、それと自動解析結果を比較した。精度を表 5 に示す。ベースラインとは、類似度に基づく格解析手法 (河原, 黒橋 2005) である。この表より、ベースラインから大幅に改善しており、提案手法が有効であることがわかる。

4.3　格フレームのカバレージ

解析における格フレームのカバレージを調べるために、格要素がその係り先用言の格フレームの用例になっているかどうかを調べた。正しい係り受けのみを評価したところ、60.7%の格要素が格フレームの用例となっていた。比較のため、新聞記事 26 年分の 2,600 万文から構築した格フレームで同様の実験を行ったところ、35.1%であった。これより、ウェブテキスト 5 億文から構築した格フレームは高いカバレージをもっていることが確認された。

また、英語の統計的構文解析器において、テスト文中の 2 項間の依存関係が学習コーパス中に存在する割合が約 1.5%であるという報告がある (Bikel 2004)。言語・リソースの違いがあるので直接の比較はできないが、格フレームのカバレージは非常に高いと思われる。

5　関連研究

これまでに、語彙的選好を明示的に扱う構文解析手法がいくつか提案されてきた。白井らは、PGLR の枠組みに基づく統計的構文解析手法を提案している (白井他 1998)。語彙的選好として、例えば P(パイ | を, 食べる) のような確率を新聞記事 5 年分から学習している。しかし、本研究で用いたような格フレームは導入しておらず、用言の意味的曖昧性を区別せずに確率推定を行っている。京都テキストコーパス中の比較的短い 500 文を用いて評価を行い、84.34%の解析精度であったと報告している。

藤尾らは、語の共起確率に基づく構文解析手法を提案している (藤尾, 松本 1999)。2 つの語が係り受けをもつ確率と距離確率の積で定義した確率モデルを用いており、それらの確率は EDR コーパスから学習している。EDR コーパス 1 万文を用いて評価を行い、86.89%であったと報告している[3]。

阿辺川らは、同じ用言を係り先とする格要素間の従属関係と、格要素・用言間の共起関係を利用した構文解析手法を提案している (阿辺川, 奥村 2006)。これら 2 つの関係を新聞記事 30 年分から収集し確率モデルを学習している。既存の構文解析器の出力する n-best の構文木候補に対して、確率モデルに基づくリランキングを適用し、もっとも確率値の高い構文木を選択している。京都テキストコーパス中の約 9,000 文を用いて評価を行い、既存の構文解析器よりも 0.26%高い 91.21% の精度を実現している[3]。さらに、阿辺川らの被連体修飾詞の解析 (阿辺川, 奥村 2005) を統合することによって、0.04%高い 91.25%の精度を得ている。

一方、語彙情報を素性として用いている様々な機械学習手法が提案されている。その中でもっとも良い精度を実現しているのは、工藤らが提案している統計的構文解析手法である (工藤, 松本 2002)。この手法は、SVM に基いてチャンキングを段階的に適応していくモデルであり、京都テキストコーパスから学習している。同コーパス (約 40,000 文) を用いて 2 分割交差検定により評価を行い、90.46%の精度を実現している[3]。しかし、数万文程度のタグ付きコーパスからでは、係り先候補間の語彙的選好を十分学習するのはほとんど困難であると思われる。なお、本論文の実験で比較対象とした「CaboCha」は、本手法を実装した解析器である。

6　おわりに

本論文では、ウェブから獲得した格フレームに基づく構文・格解析の統合的確率モデルを提案した。このモデルによって、構文解析の精度が向上することを確認した。今後は、省略・照応解析を統合することによって、格フレームに基づく構文・格・省略・照応解析の統合的確率モデルを構築する予定である。

[3] 文末から 2 つ目の文節も評価に入れている。

参考文献

Bikel, D. M. (2004). “Intricacies of Collins’ Parsing Model.” *Computational Linguistics*, **30** (4), 479–511.

藤尾正和, 松本裕治 (1999). “語の共起確率に基づく係り受け解析とその評価.” 情報処理学会論文誌, **40** (12), 4201–4212.

白井清昭, 乾健太郎, 徳永健伸, 田中穂積 (1998). “統計的構文解析における構文的統計情報と語彙的統計情報の統合について.” 自然言語処理, **5** (3), 85–106.

河原大輔, 黒橋禎夫 (2005). “格フレーム辞書の漸次的自動構築.” 自然言語処理, **12** (2), 109–132.

河原大輔, 黒橋禎夫 (2006). “高性能計算環境を用いた Web からの大規模格フレーム構築.” 情報処理学会 自然言語処理研究会 2006-NL-171, pp. 67–73.

河原大輔, 黒橋禎夫, 橋田浩一 (2002). “「関係」タグ付きコーパスの作成.” 言語処理学会 第 8 回年次大会, pp. 495–498.

工藤拓, 松本裕治 (2002). “チャンキングの段階適用による係り受け解析.” 情報処理学会論文誌, **43** (6), 1834–1842.

黒橋禎夫, 長尾眞 (1994). “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, **1** (1), 35–57.

南不二男 (1993). 現代日本語文法の輪郭. 大修館書店.

阿辺川武, 奥村学 (2005). “日本語連体修飾節と被修飾名詞間の関係の解析.” 自然言語処理, **12** (1), 107–123.

阿辺川武, 奥村学 (2006). “共起情報及び複数格の組み合わせを考慮した係り受け解析.” 自然言語処理, **13** (2), 43–62.

略歴

河原 大輔: 1997 年京都大学工学部電気工学第二学科卒業。1999 年同大学院修士課程修了。2002 年同大学院博士課程単位取得認定退学。東京大学大学院情報理工学系研究科学術研究支援員を経て、2006 年独立行政法人情報通信研究機構研究員、現在に至る。構文解析、省略解析、知識獲得の研究に従事。

黒橋 禎夫: 1989 年京都大学工学部電気工学第二学科卒業。1994 年同大学院博士課程修了。京都大学工学部助手、京都大学大学院情報学研究科講師、東京大学大学院情報理工学系研究科助教授を経て、2006 年京都大学大学院情報学研究科教授、現在に至る。自然言語処理、知識情報処理の研究に従事。

(2006 年 11 月 12 日 受付)
(2007 年 3 月 12 日 再受付)
(2007 年 4 月 13 日 採録)