Webから獲得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
東京大学 大学院情報理工学系研究科
{kawahara, kuro}@kc.t.u-tokyo.ac.jp

1 はじめに

述語項構造は、「誰がどこで何をした」のように、文章中の各用言にどのような要素がどのような関係で結びついているかを表現したものであり、文章の意味を担う基本的な単位である。このような述語項構造を認識する技術は自然言語理解において重要であり、これまで構文解析の次のステップとして位置付けられてきた。

英語では、項の統語的役割 (主格、目的格など) は、ほとんどの場合は語順から明らかである。Blaheta らは、PennTreebank を対象として統語的役割の解析を行い、95.7% という高い精度を実現している [1]。近年では、動作主、経験者のような意味的役割を明らかにする研究が盛んに行われている (e.g., [3])。

日本語においては、格助詞が統語的役割を示すものの、省略が多く、また「は」「も」などの係助詞や連体修飾によって格助詞が明示されない場合も多いため、その解析は難しい [2]。日本語において述語項構造を明らかにするためには、文法的知識だけではなく、各用言にどのような名詞がどのような関係で結びついているかを記述した知識が必要となる。本研究では、そのような知識として自動獲得した大規模格フレームを用いる。

上記のように格解析研究は活発に行われており、正解の構文構造を入力した場合には高い精度で認識できるようになってきた。しかし、自動構文解析結果を用いた場合には大幅に精度が落ちることが報告されており、構文解析が大きな問題となっている [3]。このような問題を解決するためには、構文解析と格解析を統合することが考えられる。つまり、構文構造・格構造の曖昧性を同時に考慮し、格フレームの語彙的知識に基づいて双方の曖昧性を解消することが考えられる。

本稿では、日本語を対象とした構文・格解析の統合的確率モデルを提案する。本モデルは、構文・格解析を生成的確率モデルで行い、もっとも確率値の高い構文・格構造を選択するということを行う。語彙的知識としては、できるだけ大規模なものが望ましいため、Web から獲得した大規模格フレーム [4] を用いる。

2 構文・格解析の統合的確率モデル

本稿で提案する構文・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 と格構造 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} と格構造 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。

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

2.1 構文・格解析の統合的確率モデルの概略

本研究では、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。

$$P(T, L, S) = \prod_{i=1..n} P(C_i|b_{h_i}) \quad (2)$$

n は文 S 中に存在する節の数 (=用言数) であり、b_{h_i} は節 C_i の係り先文節である。主節 C_n は係り先をもたないが、係り先を $b_{h_n} = \text{EOS}$ とする。

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。文節 b_{h_i} も同様に、語 w_{h_i} とタイプ f_{h_i} に分けて考える。

$$\begin{aligned} P(C_i|b_{h_i}) &= P(CS_i, f_i|w_{h_i}, f_{h_i}) \\ &= P(CS_i|f_i, w_{h_i}, f_{h_i})P(f_i|w_{h_i}, f_{h_i}) \\ &\approx P(CS_i|f_i, w_{h_i})P(f_i|f_{h_i}) \end{aligned} \quad (3)$$

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。

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

$P(CS_i|f_i, w_{h_i})$ を述語項構造生成確率、$P(f_i|f_{h_i})$ を用言タイプ生成確率と呼び、それぞれ 2.2、2.3 節で説明する。

2.2 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 $P(CS_i|f_i, w_{h_i})$

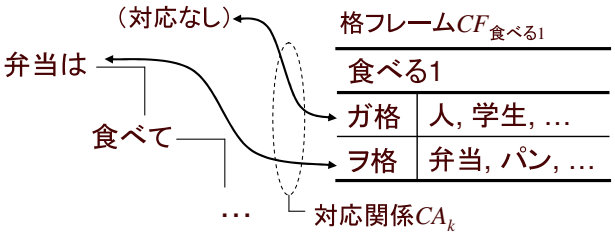

図 1: 格の対応関係 CA_k の例

は次のようになる。

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

この近似は、述語 v_i はその係り先の語 w_{h_i} 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ものである。

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。

2.2.1 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。

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

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。

$$\begin{aligned} P(A(s_j)=1, n_j, f_j | CF_l, f_i, s_j) = \\ P(A(s_j)=1 | CF_l, f_i, s_j) \\ \times \quad P(n_j, f_j | CF_l, f_i, A(s_j)=1, s_j) \end{aligned} \tag{6}$$

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び、2.2.2 節で説明する。

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。

$$\begin{aligned} P_1(CS(\text{弁当は食べる})|\ \text{テ形}, \text{出発する}) = \\ P(\text{食べる}\ |\ \text{出発する}) \\ \times \quad P(CF_{\text{食べる 1}}|\ \text{食べる}) \\ \times \quad P(\text{弁当}, \text{は}\ |CF_{\text{食べる 1}}, \text{テ形}, A(\text{を})=1, \text{を}) \\ \times \quad P(A(\text{を})=1|CF_{\text{食べる 1}}, \text{を}) \\ \times \quad P(A(\text{が})=0|CF_{\text{食べる 1}}, \text{が}) \end{aligned}$$

2.2.2 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。

$$\begin{aligned} P(n_j, f_j | CF_l, f_i, A(s_j)=1, s_j) \approx \\ P(n_j | CF_l, A(s_j)=1, s_j)\ P(f_j | s_j, f_i) \end{aligned} \tag{7}$$

$P(n_j|CF_l, A(s_j)=1, s_j)$ 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。

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

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コーパス [5] 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。

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

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

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 [6] を参考にして設定した 5 段階を考える。

2.3 用言タイプ生成確率

用言タイプ生成確率 $P(f_i|f_{h_i})$ は、文節 b_{h_i} 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_i} に係るかどうかを考慮する。

$$P_{VBmod}(f_i|f_{h_i}) = P_{VBmod}(p_i, u_i|p_{h_i}, u_{h_i}, o_{h_i}) \tag{11}$$

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。

$$P_{NBmod}(f_i|f_{h_i}) = P_{NBmod}(p_i|o_{h_i}) \tag{12}$$

3 実験

提案手法によって解析した構文・格構造の評価実験を行った。各パラメータは表 1 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 [2] である。格フレームは Web 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 Web 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。

表 1: 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	$P(p_j\|o_i, u_j)$	京都コーパス
提題助詞生成確率	$P(t_j\|o_i, u_i, p_j)$	京都コーパス
付属語列生成確率	$P(p_i, u_i\|p_{h_i}, u_{h_i}, o_{h_i})$	京都コーパス
表層格生成確率	$P(c_j\|s_j)$	京都コーパス (関係)
用言生成確率	$P(v_i\|w_{h_i})$	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

表 2: 構文解析の精度

	構文解析のみ	本手法
すべて	3447/3976 (86.7%)	3477/3976 (87.4%)
体言 → 用言	1310/1547 (84.7%)	1328/1547 (85.8%)
係助詞句	244/298 (81.9%)	242/298 (81.2%)
それ以外	1066/1249 (85.3%)	1086/1249 (86.9%)
体言 → 体言	525/556 (94.4%)	526/556 (94.6%)
用言 → 用言	593/760 (78.0%)	601/760 (79.1%)
用言 → 体言	453/497 (91.1%)	457/497 (92.0%)

構文解析実験は、Web テキスト 675 文*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 2 に示す。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。表において、「構文解析のみ」とは、格解析を伴わない構文解析器 KNP による精度である。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「構文解析のみ」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。これより本手法が、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。

以下に、「構文解析のみ」では誤りになるが、本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。

- 水が 高い× ところから低いところへ 流れる。○
- 男の空色のはずの 法衣が、 濃紺に 見える○ ほどに。×

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。Web 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。なお、係り先の誤っているものは除いて評価を行った。精度を表 3 に示す。ベースラインとは、もっとも正解となることが多かったガ格を常に解とした場合の精度である。この表より、ベースラインから大幅に改善しており、本手法が有効であることがわかる。

*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。

表 3: 格解析の精度

	ベースライン	本手法
係助詞句	58/85 (68.2%)	67/85 (78.8%)
被連体修飾詞	60/127 (47.2%)	104/127 (81.9%)

4 おわりに

本稿では、Web 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。

参考文献

[1] Don Blaheta and Eugene Charniak. Assigning function tags to parsed text. In *Proceedings of NAACL2000*, pp. 234–240, 2000.

[2] Daisuke Kawahara and Sadao Kurohashi. Fertilization of case frame dictionary for robust Japanese case analysis. In *Proceedings of COLING2002*, pp. 425–431, 2002.

[3] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. Shallow semantic parsing using support vector machines. In *Proceedings of the HLT-NAACL2004*, pp. 233–240, 2004.

[4] 河原大輔, 黒橋禎夫. 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, 2006.

[5] 河原大輔, 黒橋禎夫, 橋田浩一. 「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, 2002.

[6] 南不二男. 現代日本語文法の輪郭. 大修館書店, 1993.