文末からのトップダウン係り受け解析との同時実行に基づく日本語文の語順整序と読点挿入

宮地 航太 †1　大野 誠寛 †2　松原 茂樹 †1
†1 名古屋大学　†2 東京電機大学
miyachi.kota@j.mbox.nagoya-u.ac.jp

1 まえがき

日本語は，語順が比較的自由であるとされているが，実際には語順に関して選好が存在している．そのため，文法的には間違っていないものの読みにくい語順を持った文が作成されることがある．また読点についても同様に，読みやすい文を作成するためには適切な位置に読点を打つ必要がある．例えば，以下の**文 1** は読みにくいが，**文 2** のように文節を並べ替え，読点を挿入すれば読みやすくなる [1]．

文 1　私は家を都会に憧れ出た．
文 2　私は，都会に憧れ家を出た．

語順整序や読点挿入に関する研究は，推敲支援や文生成などに応用でき，いくつか行われている [2–15]．その中でも，語順や読点と係り受けとの相互依存的な関係に着目し，係り受け解析と語順整序や読点挿入を同時実行する手法が存在する．大野ら [16] は係り受け解析と語順整序の同時実行を実現している．ただし，大野らの手法では読点挿入を対象としていない．これに対し，宮地ら [17] は，Shift-Reduce アルゴリズムを拡張することにより，係り受け解析と語順整序に読点挿入を加えた 3 つの処理を同時実行する手法を提案している．しかし，宮地らの手法では文頭から局所的に解析を行っているため，2 文節の語順を決定する際に各文節の係り先の情報をほとんど用いることが出来ておらず，その精度は十分とはいえない．

そこで本論文では，推敲支援のための要素技術として，読みにくい語順をもった日本語文に対して，文末からトップダウンに，係り受け構造と語順，読点位置を同時的に決定する手法を提案する．本手法では，係り受け解析と語順整序，読点挿入を同時実行する対象を入力文の文末から各文節までの部分文節列とすることにより，語順の判断材料として各文節の係り先の情報を利用できるようにする．これは，読みにくい入力文であっても非文でなければ，係り受けの後方修飾性を満たすことから，各文節の係り先がその対象部分文節列内に存在するためである．また，本手法は，各部分文節列に対する係り受け構造と語順，読点位置を 2 分木で表現することにより，そのあらゆる組合せを簡潔に探索する．

2 先行研究

係り受け解析は一般に，入力文の語順や読点が適切でない場合，精度が低下する [3, 16, 18]．一方，読みやすい語順や読点とするために最初に語順整序や読点挿入をそれぞれ単独で実行すると，係り受け情報が利用できず，それらの精度は低下すると考えられる．また，語順が変われば，適切な読点位置も変わる [3]．このように係り受けと語順，読点位置は互いに依存しているといえる．従って，入力文の語順と読点を読みやすく整形するという問題に対するアプローチとしては，係り受け解析，語順整序，読点挿入を同時実行する手法が有望である．

この考えに基づき，宮地ら [17] はこれら 3 つの処理の同時実行手法を提案している．この手法は，Shift-Reduce アルゴリズムを拡張し，入力文中の局所的な 2 文節に着目して，それらの間の係り受け関係の有無や語順，読点有無を同時的に決めることを文頭から繰り返す．一般に語順整序は，係り受け構造が付与されていることを前提に，係り受けの非交差性と後方修飾性を保つため，同じ係り先を持つ文節同士において行われ，その係り先の情報を利用する [2]．しかし宮地らの手法では，文頭から解析を進め，対象となる 2 文節の間で考えられる係り受け関係有無，語順，読点有無の組合せのみを候補として検討するため，それらの係り先が同じである否かを考慮しない．そのため語順整序の性能が損なわれていると考えられる．

そこで本研究では，文末からのトップダウン係り

受け解析を拡張し，文末から各文節までの部分文節列に対して係り受け構造，語順，読点位置の 3 項組からなる構造を同定することを，文頭まで繰り返すという戦略を採用する．文末から各文節までの部分文節列は，後方修飾性から必ず閉じた係り受け構造をもつため，考えられる係り受け構造の各候補上で同じ係り先を持つ文節同士だけを語順入替の対象とすることが自然に可能となる．

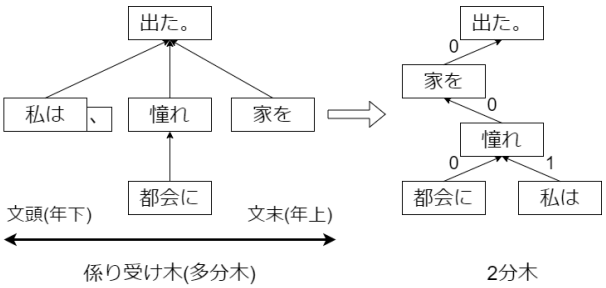

図 1　2 分木での表現例

3　提案手法

本手法では，意味は伝わるものの読みにくい語順を持った文が入力されることを想定し，その文に対して，係り受け解析を行うと同時に，読みやすい語順と読点位置を同定する．入力文の文末から順に解析していき，文末から各文節までの部分文節列における係り受け構造と語順，読点位置を決定することを繰り返す，という戦略により，1 文に対する係り受け解析，語順整序，読点挿入の同時実行を実現する．本節では，まず 3.1 節で，入力文に対して係り受け構造，読みやすい語順と読点位置を決定するアルゴリズムについて述べる．次に，A 節では，アルゴリズムの中で確率を計算する際に用いる確率モデルについて述べる．

3.1　アルゴリズム

本手法では，係り受け構造と語順，読点位置を 2 分木により表現することにより，そのあらゆるパターンを効率的に探索する．その 2 文木は次のように作る．

- 各ノードに文節を割当てる．
- 係り受け関係のあるノード間をエッジで結び，各エッジにはその下側のノードの直後に読点がある [1] か否か [0] を示すラベルを持たせる．
- 各ノードの左の子には，そのノードの文節に係る文節のうち，最も文末に近い文節（係り受け木における長子）を配置する．
- 各ノードの右の子には，そのノードの文節と同じ係り先を持つ文節のうち，そのノードの直前に位置する文節（係り受け木における直弟）を配置する．なお，係り受け木において兄弟関係にある（すなわち，係り先が同じ）文節間の上下関係は，文末側に位置するほど年上であるとする．

例えば，1 節で示した**文 2** における係り受け構造と語順，読点位置を 2 分木で表現すると図 1 のようになる．

本手法では，次の手順で入力文の文節列を末尾から順に処理する．

1. 入力文節列を入力語順でキューに格納する．
2. キューから文末文節を取り出して 2 分木の根とする．更に 1 つ文節を取り出し，根の左の子の位置に挿入し，根との間のエッジの読点有無ラベル（すなわち，この文節の直後の読点有無）を同定することにより，2 分木を生成する．
3. キューから 1 つ文節を取り出し，その文節を既に構築済みの 2 分木に挿入する形で新たな 2 分木を生成する．その際，既に構築済みの 2 分木を前提として，係り受けの構文的制約上，挿入可能な位置や，新たに読点有無ラベルを判定すべきエッジのラベル値の組合せを考え，それらを新たな 2 分木の候補とし，最適な 2 分木を選択する．この候補の生成については 3.1.1 で述べる．また，この候補選択のための確率モデルについては付録 A に示す．
4. 3 を繰り返し，キューが空になれば終了する．

3.1.1　係り受け構造，語順，読点位置の候補

前述した手順 3 における 2 分木の候補は，まず，既存の 2 分木の中で，新たにキューから取り出した文節を挿入できる位置を考え，次に，それらの各位置に挿入した各 2 分木において，新たに読点有無ラベルを判定する必要があるエッジを考えることにより生成する．

文節列 $B_{i+1:n} = b_{i+1} \cdots b_n$ に対する 2 分木に対して，新たな文節 b_i を挿入できる位置は，入力語順でも出力語順でも係り受けの後方修飾性と非交差性を共に満たすという制約に基づき決まるが，2 分木の性質から簡単に調べ上げることができる．具体的には，次の位置に限られる．

- 直前に挿入されたノード b_{i+1} の左右の子
- b_{i+1} から根 b_n に至る経路上の各エッジ
- b_{i+1} から根 b_n に至る経路上の各ノード（b_n は除く）から，右の子のみを辿って右の子を持たないノードに行きつくまでの各エッジと，その行きついたノードの右の子

なお，あるエッジに b_i を挿入するとは，そのエッジの両端にあるノードの間に新たなノード b_i を挿入することを意味する．

次に，上述の各位置に b_i を挿入した 2 分木の各々において，新たに読点有無ラベルを判定する必要があるエッジとは，

- 新たに挿入したノード b_i から上下に延びる各エッジ （ただし，子を持たない場合は上に延びるエッジのみ）
- b_i から右の子のみを辿って，右の子を持たないノードに至る経路上の各エッジ，
- b_i から根 b_n に至る経路上の各ノード（b_i と係り受け木において兄弟関係にあるノードと，b_n は除く）から，右の子のみを辿って右の子を持たないノードに行きつくまでの各エッジ

である．これらは，文節 b_i の直前直後，及び，b_i の挿入によって係り受け距離が伸びる各文節の直後を表す．ただし，既に読点有と判定されているエッジはそのままとする．

3.2 アルゴリズムの動作例

図 2 に 1 節の**文 1** を入力として与えられ，**文 2** を出力するときのアルゴリズムの動作例を示す．ステップ 1 では入力文「私は家を都会に憧れ飛び出した。」が文節ごとにキューに格納される．ステップ 2 ではキューから文末文節「出た。」が取り出され根とし，文節「憧れ」が取り出され根の左の子とする．またこれらを結ぶエッジの読点有無を同定する．ステップ 3-a では文節「都会に」がデキューされる．このとき既存の 2 分木に対して「都会に」が挿入される位置の候補は 3 箇所存在する．各位置に挿入した 2 分木において読点有無を新たに判定するエッジのラベルが?で示されている．「都会に」は「憧れ」に係るため，①が選択され，また「都会に」と「憧れ」の間には読点は打たれない．ステップ 3-b では文節「家を」がデキューされる．このとき 2 分木に対して「家を」を挿入できる位置は 5 箇所存在する．「家を」は「出た。」に係り，また，適切な語順は「憧れ」の後であるため⑤が選択され，⑤に挿入した場合の 2 分木においてラベルが?となっているエッジについて読点有無を同定し，今回はいずれも読点無となる．ステップ 3-c では文節「私は」がデキューされる．このとき既存の 2 分木に対して「私は」を挿入できる位置は 4 箇所存在する．「私は」は「出た。」に係り，また，適切な語順は先頭であるため②が選択される，②に挿入した場合の 2 分木においてラベルが?となっているエッジについて読点有無を同定し，今回は「私は」と「都会に」を結ぶエッジ（「私は」の直後）に読点が打たれる．キューが空になったためステップ 4 でアルゴリズムが終了し，1 節の**文 2** が出力される．

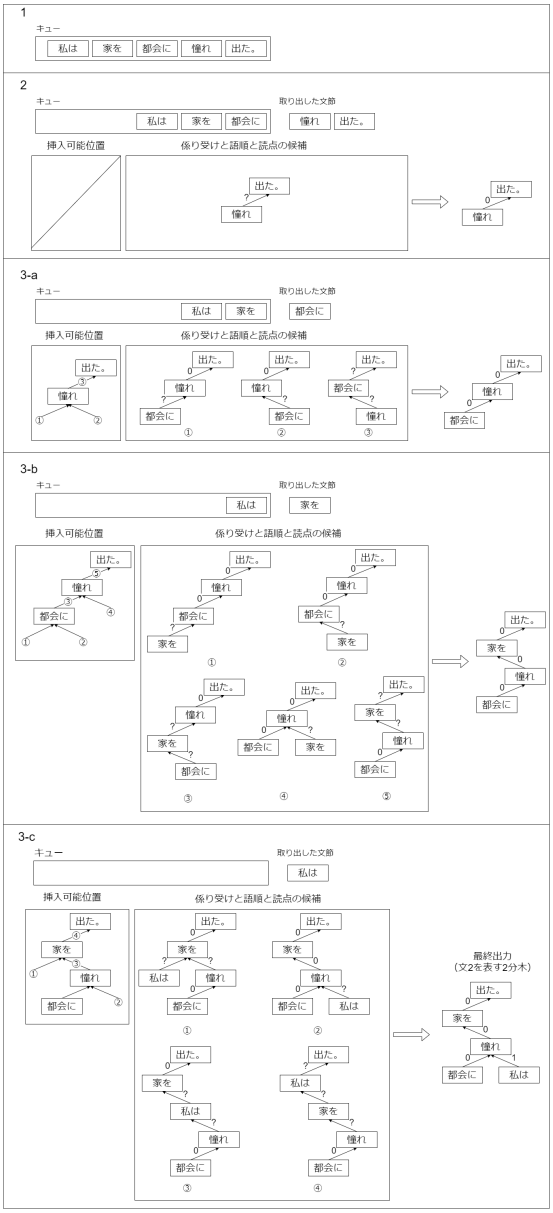

図 2　アルゴリズムの動作例

4 評価実験

読みにくい日本語文の語順整序および読点挿入における本手法の性能を評価するため，新聞記事を用いた実験を実施した．新聞記事中の文から擬似的に作成した読みにくい語順の文に対して本手法を適用し，元の文の語順と読点をどの程度再現できるかにより評価した．

4.1 実験概要

評価用データには宮地ら [17] と同じ手順で作成された読点付きの読みにくい語順の文データ 1,000 文を用いた．A.1 の各確率を推定するための機械学習モデルには勾配ブースティングマシン（GBM）を採用し，そのツールとして LightGBM[1] を用い，パラメータチューニングには Optuna[2] を使用した．学習には，京大テキストコーパス Ver.4.0 [19] のうち，評価用データの作成に用いた文を除く 35,404 文を用いた．

語順整序の評価では，文献 [16] と同様に，文単位正解率（語順整序後の語順が元の文と完全に一致している文の割合）と 2 文節単位正解率（2 文節ずつ取り上げたときの文節の順序関係が元の文のそれと一致しているものの割合）を測定した．読点挿入の評価では，正解文と語順が完全一致している文のみを対象として，文献 [18] と同様に，京大コーパスの読点を正解とした場合の再現率と適合率を測定した．

比較手法には宮地ら [17] の手法を用意した．

4.2 実験結果

実験結果を表 1 に示す．提案手法は，比較手法と比べて，語順整序の 2 文節単位正解率や読点の再現率，適合率，F 値において高い値を達成しており，本手法の有効性が確認された．本手法による語順整序及び読点挿入が正解と完全に一致した例を図 3 に示す．読みにくい語順を持った入力文に対して，読みやすい語順に修正した上で正しく読点が挿入できている．

表 1　実験結果

	語順		読点		
	2 文節単位	文単位	再現率	適合率	F 値
比較手法	64.50% (20,486/ 31,760)	9.60% (96/ 1,000)	36.17% (17/47)	58.62% (17/29)	44.74
提案手法	75.73% (24,051/ 31,760)	6.50% (65/ 1,000)	65.33% (49/75)	58.62% (49/65)	70.00

入力文：

独身時代は自分の手帳を持ち、びっしりと、仕事の日程やおけいこごと、テニススクール、食事会の予定などが書き込まれていた。

比較手法：

独身時代は仕事の持ち、おけいこごと、手帳をびっしりと日程やテニススクール自分の食事会の予定などが書き込まれていた。

提案手法（正解）：

独身時代は自分の手帳を持ち、仕事の日程やおけいこごと、テニススクール、食事会の予定などがびっしりと書き込まれていた。

図 3　語順整序及び読点挿入の成功例

5 まとめ

本論文では，読みにくい語順の文に対して係り受け解析，語順整序，読点挿入を文末からトップダウンに同時実行する手法を提案した．評価実験の結果，本手法の有効性を確認した．今後は，確率の推定に用いる素性や機械学習手法を見直すことにより，精度向上を図りたい．

謝辞　本研究は，一部，科研費 No. 26280082，No. 19K12127 により実施した．

1） https://lightgbm.readthedocs.io/en/latest/
2） https://github.com/optuna/optuna

参考文献

[1] 日本語記述文法研究会. 現代日本語文法 7. くろしお出版, 2009.

[2] 内元清貴, 村田真樹, 馬青, 関根聡, 井佐原均. コーパスからの語順の学習. 自然言語処理, Vol. 7, No. 4, pp. 163–180, 2000.

[3] 村田匡輝, 大野誠寛, 松原茂樹. 読点の用法的分類に基づく日本語テキストへの自動読点挿入. 電子情報通信学会論文誌, Vol. J95-D, No. 9, pp. 1783–1793, 2012.

[4] Allen Schmaltz, Alexander M Rush, and Stuart M Shieber. Word ordering without syntax. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 2319–2324, 2016.

[5] 横林博, 菅沼明, 谷口倫一郎ほか. 係り受けの複雑さの指標に基づく文の書き換え候補の生成と推敲支援への応用. 情報処理学会論文誌, Vol. 45, No. 5, pp. 1451–1459, 2004.

[6] Katja Filippova and Michael Strube. Generating constituent order in german clauses. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 320–327, 2007.

[7] Karin Harbusch, Gerard Kempen, Camiel Van Breugel, and Ulrich Koch. A generation-oriented workbench for performance grammar: Capturing linear order variability in german and dutch. In *Proceedings of the 4th International Natural Language Generation Conference*, pp. 9–11, 2006.

[8] Geert-Jan M Kruijff, Ivana Kruijff-Korbayová, John Bateman, and Elke Teich. Linear order as higher-level decision: Information structure in strategic and tactical generation. In *Proceedings of 8th European Workshop on Natural Language Generation*, pp. 74–83, 2001.

[9] Eric Ringger, Michael Gamon, Robert C Moore, David M Rojas, Martine Smets, and Simon Corston-Oliver. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th International Conference on Computational Linguistics*, pp. 673–679, 2004.

[10] James Shaw and Vasileios Hatzivassiloglou. Ordering among premodifiers. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 135–143, 1999.

[11] 鈴木英二, 島田静雄, 近藤邦雄, 佐藤尚ほか. 日本語文章における句読点自動最適配置. 情報処理学会全国大会講演論文集, Vol. 50, No. 3, pp. 185–186, 1995.

[12] 熊野明, 吉村裕美子, 野上宏康ほか. 自然な日本語生成のための指針. 情報処理学会全国大会講演論文集, Vol. 41, No. 3, pp. 165–166, 1990.

[13] Agustin Gravano, Martin Jansche, and Michiel Bacchiani. Restoring punctuation and capitalization in transcribed speech. In *Proceedings of 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4741–4744, 2009.

[14] Wei Lu and Hwee Tou Ng. Better punctuation prediction with dynamic conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 177–186, 2010.

[15] Yuqing Guo, Haifeng Wang, and Josef Van Genabith. A linguistically inspired statistical model for chinese punctuation generation. *ACM Transactions on Asian Language Information Processing*, Vol. 9, No. 2, pp. 1–27, 2010.

[16] 大野誠寛, 吉田和史, 加藤芳秀, 松原茂樹. 係り受け解析との同時実行に基づく日本語文の語順整序. 電子情報通信学会論文誌, Vol. J99-D, No. 2, pp. 201–213, 2016.

[17] 宮地航太, 大野誠寛, 松原茂樹. 係り受け解析との同時実行に基づく日本語文の語順整序と読点挿入. 言語処理学会第 26 回年次大会発表論文集, pp. 243–246, 2020.

[18] 宮地航太, 大野誠寛, 松原茂樹. 読みにくい語順の文への読点の自動挿入. 言語処理学会第 25 回年次大会発表論文集, pp. 1308–1311, 2019.

[19] 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. 言語処理学会第 3 回年次大会論文集, pp. 115–118, 1997.

[20] 内元清貴, 関根聡, 井佐原均. 最大エントロピー法に基づくモデルを用いた日本語係り受け解析. 情報処理学会論文誌, Vol. 40, No. 9, pp. 3397–3407, 1999.

A　候補選択のための確率モデル

入力文の文節列を $B = b_1, \cdots, b_n$ とし，3.1 節のアルゴリズムのステップ 3 において，文節 $b_i (1 \leq i < n-1)$ がデキューされたとする．このとき，文節列 $B_{i:n} = b_i, \cdots, b_n$ に対する 2 分木 $S_{i:n}$ を $argmax P(S_{i:n}|B)$ により決定する．ここで，$S_{i:n}$ は，語順整序後の語順 $O_{i:n} = \{o_{i,i+1}, o_{i,i+2}, \cdots, o_{i:n-1}, o_{i+1,i+2}, \cdots, o_{x,y}, \cdots, o_{n-2,n-1}\}$，読点位置 $C_{i:n} = \{c_i, c_{i+1}, \cdots, c_x, \cdots, c_{n-1}\}$，係り受け構造 $D_{i:n} = \{d_i, d_{i+1}, \cdots, d_x, \cdots, d_{n-1}\}$ の三項組として定義され，$S_{i:n} = \langle O_{i:n}, C_{i:n}, D_{i:n} \rangle$ と書く．ここで $o_{x,y} (i \leqq x < y < n)$ は，文節 b_x と b_y の間の順序関係を表し，b_x が b_y より文頭側に位置するか（$o_{x,y} = 1$），否か（$o_{x,y} = 0$）の 2 値をとる．また $c_x (i \leqq x < n)$ は，文節 b_x の直後に読点が有るか（$c_x = 1$），無いか（$c_x = 1$）の 2 値の値をとる．最後に $d_x (i \leqq x < n)$ は，文節 b_x を係り元とする係り受け関係とする．

A.1　確率モデル

ある $S_{i:n} = \langle O_{i:n}, C_{i:n}, D_{i:n} \rangle$ に対する $P(S_{i:n}|B)$ を，下式により計算する．

$$
\begin{aligned}
&P(S_{i:n}|B) = P(O_{i:n}, C_{i:n}, D_{i:n}|B) \\
&= \sqrt[3]{P(O_{i:n}|B) * P(D_{i:n}|B, O_{i:n}) * P(C_{i:n}|B, O_{i:n}, D_{i:n})} \\
&\times \sqrt[3]{P(D_{i:n}|B) * P(O_{i:n}|B, D_{i:n}) * P(C_{i:n}|B, O_{i:n}, D_{i:n})} \\
&\times \sqrt[3]{P(O_{i:n}|B) * P(C_{i:n}|B, O_{i:n}) * P(D_{i:n}|B, O_{i:n}, C_{i:n})}
\end{aligned}
$$

上式の最右辺における各確率は，2 文節間の語順 $o_{x,y}$ は他の 2 文節間の語順とは互いに独立であり，かつ，係り受け関係 d_x も他の係り受け関係とは独立であり，かつ，読点位置 c_x は直後の読点位置を除く，他の読点位置とは独立であると仮定すると，以下のように近似できる．

$$P(O_{i:n}|B) \fallingdotseq \prod_{x=i}^{n-2} \prod_{y=x+1}^{n-1} P(o_{x,y}|B)$$

$$P(D_{i:n}|B, O_{i:n}) \fallingdotseq \prod_{k=i}^{n-1} P(d_k|B, O_{i:n})$$

$$P(C_{i:n}|B, O_{i:n}, D_{i:n}) \fallingdotseq \prod_{k=1}^{n-i} P(c_{n-k}|B, O_{i:n}, D_{i:n}, C_{n-k+1:n})$$

$$P(D_{i:n}|B) \fallingdotseq \prod_{k=i}^{n-1} P(d_k|B)$$

$$P(O_{i:n}|B, D_{i:n}) \fallingdotseq \prod_{x=i}^{n-2} \prod_{y=x+1}^{n-1} P(o_{x,y}|B, D_{i:n})$$

$$P(C_{i:n}|B, O_{i:n}) \fallingdotseq \prod_{k=1}^{n-i} P(c_{n-k}|B, O_{i:n}, C_{n-k+1:n})$$

$$P(D_{i:n}|B, O_{i:n}, C_{i:n}) \fallingdotseq \prod_{k=i}^{n-1} P(d_k|B, O_{i:n}, C_{i:n})$$

A.2　機械学習に用いる素性

$P(o_{x,y}|B, D_{i:n})$ を推定する際には，文献 [16] で語順を推定する際に用いられた素性のうち，読点に関する素性を除く全ての素性を用いる．$P(o_{x,y}|B)$ の推定では，$P(o_{x,y}|B, D_{i:n})$ の推定時に使用した素性のうち，係り受け情報を使うことなく取得可能な素性を用いる．$P(d_{x,y}|B, O_{i:n}, C_{i:n})$ を推定する際には，文献 [20] で係り受けを推定する際に用いられた素性のうち，括弧に関する素性を除く全ての素性を用いる．$P(d_{x,y}|B, O_{i:n})$ の推定では，$P(d_{x,y}|B, O_{i:n}, C_{i:n})$ の推定時に使用した素性のうち，読点に関する情報を使うことなく取得可能な素性を用いる．$P(d_{x,y}|B)$ の推定では，$P(d_{x,y}|B, O_{i:n})$ の推定時に使用した素性のうち，語順に関する情報を使うことなく取得可能な素性を用いる．$P(c_{x,y}|B, O_{i:n}, D_{i:n})$ を推定する際には，文献 [3] で用いられた素性のうち，推定する文節境界より文頭側の係り受け情報と読点に関する情報を使うことなく取得可能な全ての素性を用いる．$P(c_{x,y}|B, O_{i:n})$ の推定では，$P(c_{x,y}|B, O_{i:n}, D_{i:n})$ の推定時に使用した素性のうち，係り受け情報を使うことなく取得可能な素性を用いる．