ベイズ学習による木接合文法獲得

進藤 裕之　　藤野 昭典　　永田 昌明
日本電信電話株式会社 NTT コミュニケーション科学基礎研究所
{shindo.hiroyuki, fujino.akinori, nagata.masaaki}@lab.ntt.co.jp

1　はじめに

木接合文法（TAG: Tree Adjoining Grammars）[6]は，計算言語学や自然言語処理，バイオインフォマティクスなどの分野で広く研究されている形式文法である．TAG は，文脈自由文法（CFG: Context Free Grammars）よりも表現能力が高いことが知られており，例えばオランダ語にみられるような交差する依存関係（cross-serial dependencies）などの言語現象をうまく捉えることができる．そのため，機械翻訳や semantic role labeling などのタスクに応用され一定の成功を収めている [4, 7].

TAG では，部分木を構成要素とし，それらを置換（substitution）または接合（adjunction）と呼ばれる操作で結合していくことにより全体の構文木を生成する．置換を行う部分木を初期木（initial tree），接合を行う部分木を補助木（auxiliary tree）と呼び，初期木と補助木をまとめて基本木（elementary tree）と呼ぶ．図 1，2，3 にそれぞれ基本木，置換操作，接合操作の例を示す．初期木，補助木ともに非終端シンボルまたは終端シンボルのノードで構成される部分木であるが，補助木は足（foot）と呼ばれる特殊な葉ノードを持っており，根（root）ノードと足ノードは必ず同じシンボルとなっていなければならない．また，基本木において根ノードでも葉ノードでもないノードを内部（internal）ノードと呼ぶ．

言語処理分野では，コーパスから統計的な手法で TAG を獲得するために，Penn Treebank などの構文木データが用いられてきた．TAG の獲得には，各構文木を構成する基本木の集合と，その生成に用いられる置換・接合操作の情報を推定する必要がある（両者を合わせて導出過程と呼ぶ）．しかし，各構文木の導出過程には複数の可能性が考えられるため，コーパスから教師なし学習で推定しなければならない．

従来研究では，尤もらしい TAG の導出過程を推定するために，主辞や付加詞などの情報に基づいた発見的手法によるものや，EM アルゴリズムなどの最尤推定を用いるものが一般的であった [1]．これらの方法は基本木の獲得に用いる言語知識に推定精度が大きく依存し，データに過学習しやすいという問題点がある [10]．

一方で，近年 CFG を拡張した木置換文法（TSG: Tree Substitution Grammars）を統計的に自動獲得する方法が盛んに研究されている．TSG は，TAG の接合操作を禁止したもので，部分木の置換操作のみで構文木を生成する枠組みである．例えば，[3, 10] の方法は，ノンパラメトリックベイズモデルを利用し，データを正確にモデル化するとともに，できるだけ少量（コンパクト）な TSG の部分木を獲得することに成功している．コンパクトな部分木の集合は，構文解析の曖昧性を低減させ，メモリ削減や学習の高速化にもつながる．

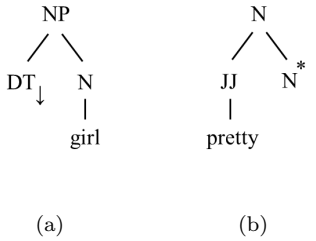

図 1　基本木の例．(a) 初期木．初期木の非終端ノードには記号“↓”が付与され，置換操作が実行されることを表す．(b) 補助木．記号“*”の付与されたノードは足ノードを表す．

今回，我々は TSG に接合操作を加えて TAG とすることで，よりコンパクトな部分木を獲得できることを期待する．また，接合操作により獲得される補助木は，初期木とは異なるタイプの言語現象を捉えられることが期待され，構文情報を用いる様々なタスクへ応用できる可能性がある．したがって，本稿では Cohn らの TSG 手法をさらに発展させ，ノンパラメトリックベイズモデルを用いて TAG の基本木を獲得するための手法を提案する．

2　確率モデル

構文木 t が与えられたとき，それを構成する基本木の集合 $\mathbf{e}$ の事後確率 $p(\mathbf{e}|t)$ は，ベイズの定理を用いて以下のように計算できる．

$$p(\mathbf{e}|t) \propto p(t|\mathbf{e})\,p(\mathbf{e}) \tag{1}$$

ここで，基本木 $\mathbf{e}$ の組み合わせによって得られる構文木が t と一致するとき $p(t|\mathbf{e}) = 1$ となり，そうで

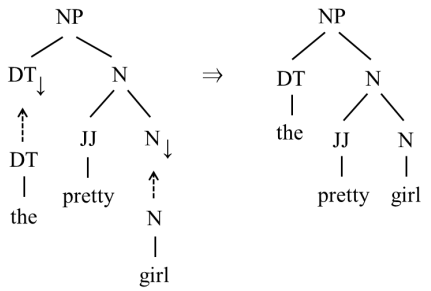

図 2　置換操作の例

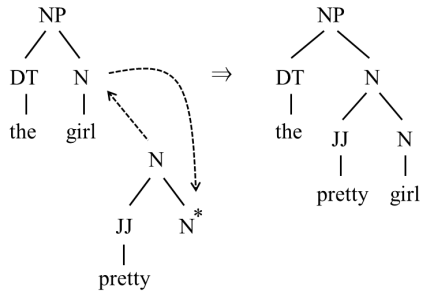

図 3　接合操作の例

ない場合は $p(t|\mathbf{e}) = 0$ となる．基本木の確率モデル $p(\mathbf{e})$ は，事前分布として Pitman-Yor 過程 [9] を仮定すると，以下のようになる．

$$p(e_i|\mathbf{e}_{-i}, X, d_X, \theta_X) = \alpha_{e_i,X} + \beta_X P_0(e_i|X) \quad (2)$$

ただし，$\mathbf{e}_{-i} = e_1, \ldots, e_{i-1}$ は，1 回目から $i-1$ 回目までに生成された基本木の集合である．X は基本木 e_i のルートシンボルを表す．また，$\alpha_{e_i,X} = \frac{n^{-i}_{e_i,X} - d_X \cdot t_{e_i,X}}{\theta_X + n^{-i}_{\cdot,X}}, \beta_X = \frac{\theta_X + d_X \cdot t_{\cdot,X}}{\theta_X + n^{-i}_{\cdot,X}}$ である．$n^{-i}_{e_i,X}$ は，$\mathbf{e}_{-i}$ のうち e_i と同じ基本木が何回生成されたかを表す．

Pitman-Yor 過程に基づく確率モデルは，内部で各基本木が何回生成されたのかという情報を，いくつかのクラスタに分けて保存している．例えば，ある基本木がこれまでに 10 回生成されたとすると，この確率モデルの内部では，(3 回，7 回）という二つのクラスタになって保持されている場合もあれば，(2 回，3 回，5 回）のように三つのクラスタになっている場合もある．このとき，$t_{e_i,X}$ は，基本木 e_i がモデル内部でいくつのクラスタに分割されているかを表す．また，$n^{-i}_{\cdot,X} = \sum_e n^{-i}_{e,X}, t_{\cdot,X} = \sum_e t_{e,X}$ である．

ここでは，初期木および補助木の確率分布をルートシンボルごとにそれぞれ独立に式 2 で定義する．以降，初期木の確率分布のパラメータおよび変数を $\{d_X, \theta_X\}$，補助木の確率分布のパラメータを $\{d'_X, \theta'_X\}$ と区別する．同様に，式 2 の変数をそれぞれ $\{\alpha_{e_i,X}, \beta_X\}$，$\{\alpha'_{e_i,X}, \beta'_X\}$ と区別する．

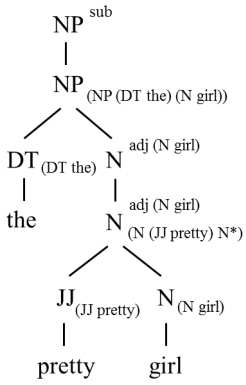

図 4　基本木分解のために変換された図 3 の導出過程

TAG では，全ての葉ノードが終端シンボルになったときに構文木の生成が終了する．したがって，初期木による置換操作は，葉ノードが非終端シンボルであれば確率 1 で必ず実行される．一方で，補助木による接合操作は内部ノードで起こるため，実行される場合とそうでない場合とがある．そこで，シンボル X を持つ内部ノードで接合操作が起こる確率を接合確率 a_X として新たに導入する．したがって，補助木 e_i が内部ノード X に対して接合される確率は，$a_X \times p(e_i|\mathbf{e}_{-i}, X, d'_X, \theta'_X)$ となる．

$P_0(e_i|X)$ は基本木 e_i の基底確率であり，[3] と同様に，基本木を CFG 規則（深さが 1 の部分木）へ分解し，各生成規則の確率の積で定義する．すなわち，

$$P_0(e_i|X) = \prod_{r \in \mathrm{CFG}(e_i)} P_{\mathrm{MLE}}(r) \times \prod_{A \in \mathrm{LEAF}(e_i)} s_A \times \prod_{B \in \mathrm{INTERNAL}(e_i)} (1 - s_B) \quad (3)$$

ただし，$\mathrm{CFG}(e_i)$ は，e_i を深さが 1 の部分木に分解した生成規則の集合で，$P_{\mathrm{MLE}}(r)$ は生成規則 r の最尤推定量を表す．また，$\mathrm{LEAF}(e_i)$，$\mathrm{INTERNAL}(e_i)$ はそれぞれ，e_i の葉ノードおよび中間ノードの集合を表す．s_X は停止確率と呼ばれ，初期木または補助木の生成がノード X で停止する確率である．

式 2 は，少数の基本木を繰り返し利用し，まれに新たな基本木を生成するモデルである．したがって，構文木コーパスからコンパクトな部分木の集合を学習することができる．

3　学習

3.1　基本木分解

式 2 の確率分布にしたがって基本木を生成するために，TAG の基本木分解という方法を提案する．これは，後述するマルコフ連鎖モンテカルロ法による部分木のサンプリングを効率的に実行するために必要な前

分解された基本木		確率
$\mathrm{NP}^{\mathrm{sub}}$	$\rightarrow \mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ girl)})}$	$\alpha_{(\mathrm{NP\ (DT\ the)\ (N\ girl)}),\mathrm{NP}}$
$\mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ girl)})}$	$\rightarrow \mathrm{DT}_{(\mathrm{DT\ the})}\mathrm{N}^{\mathrm{adj\ (N\ girl)}}$	$(1-a_{\mathrm{DT}}) \times a_{\mathrm{N}}$
$\mathrm{DT}_{(\mathrm{DT\ the})}$	$\rightarrow$ the	1
$\mathrm{N}^{\mathrm{adj\ (N\ girl)}}$	$\rightarrow \mathrm{N}^{\mathrm{adj\ (N\ girl)}}_{(\mathrm{N\ (JJ\ pretty)\ N*})}$	$\alpha'_{(\mathrm{N\ (JJ\ pretty)\ N*}),\mathrm{N}}$
$\mathrm{N}^{\mathrm{adj\ (N\ girl)}}_{(\mathrm{N\ (JJ\ pretty)\ N*})}$	$\rightarrow \mathrm{JJ}_{(\mathrm{JJ\ pretty})}\mathrm{N}_{(\mathrm{N\ girl})}$	$(1-a_{\mathrm{JJ}}) \times 1$
$\mathrm{JJ}_{(\mathrm{JJ\ pretty})}$	$\rightarrow$ pretty	1
$\mathrm{N}_{(\mathrm{N\ girl})}$	$\rightarrow$ girl	1

表 1　基本木分解の例

処理である．[3] では，TSG に対して同様の方法を提案しているが，ここではそれを整理し，さらに TAG のための接合操作を加えたものを新たに提案する．

基本木分解では，式 2 からの基本木の生成が式 2 の第一項目（$\alpha_{e_i,X}$）からの生成なのか，または第二項目（$\beta_X P_0(e_i|X)$）からの生成なのかを明示的に区別する．第一項目からの生成は，以前生成した基本木 $\mathbf{e}_{-i}$ の中から，確率的にいずれかを選択することに相当する．一方で，第二項目からの生成は，基底分布 $P_0(e_i|X)$ にしたがって新たに基本木を生成することに相当する．

まず，図 3 の導出過程は，図 4 のように変換することができる．図 4 で，$\mathrm{NP}^{\mathrm{sub}}$ は，NP で置換操作が実行されたことを表し，$\mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ girl)})}$ は基本木 (NP (DT the) (N girl)) を必ず生成するノード NP である．したがって，NP に対して初期木 (NP (DT the) (N girl)) による置換操作が実行されたと解釈することができる．また，$\mathrm{N}^{\mathrm{adj\ (N\ girl)}}$ は，内部ノード $\mathrm{N}_{(\mathrm{N\ girl})}$ で接合操作が実行されたことを表し，$\mathrm{N}^{\mathrm{adj\ (N\ girl)}}_{(\mathrm{N\ (JJ\ pretty)\ N*})}$ は $\mathrm{N}_{(\mathrm{N\ girl})}$ での接合操作が補助木 (N (JJ pretty) N*) によって実行されたことを表す．このとき，図 4 の各生成規則に対して表 1 のように確率を割り当てると，式 2 の確率にしたがって基本木を生成することができる．図 4 の例では，基本木は全て第一項目から生成されているが，第二項目から生成する場合は $\alpha_{e_i,X}$ や $\alpha'_{e_i,X}$ の代わりに $\beta_{e_i,X}$ や $\beta'_{e_i,X}$ を割り当てればよい．以上のように，TAG の導出過程を図 4 のように変換し，基本木の生成を表 1 のように CFG 規則の積で表現する方法が基本木分解である．

3.2　ブロック化 Metroporis-Hastings サンプリング

確率分布のパラメータを学習するために，マルコフ連鎖モンテカルロ法の一種であるブロック化 Metroporis-Hastings サンプリング法 [5, 2] を用いる．この方法は，各構文木ごとに以下の 3 ステップを実行する．これを何度も繰り返すことで，基本木の事後分布にしたがったサンプルを得ることができる．

1. 基本木分解された確率を用いて構文木の内側確率を動的計画法で計算
2. 1 で計算された内側確率に基づいてトップダウンに基本木をサンプリング
3. Metroporis-Hastings テストを用いてサンプルを受理または棄却

詳細は，[2] に記述されている．確率モデルの学習後，構文木が未知の文に対して構文解析を行う場合は，同様に上記の 3 ステップを実行して得られた基本木のサンプルを使用すればよい．ただし，学習のときは観測される構文木の部分木となるような基本木のみを内側確率の計算に用いるが，構文解析のときはすべての基本木を内側確率の計算に用いる．

method	(b_1, b_2)	# rules (# aux. trees)	F1
CFG	-	5957 (-)	64.99
TSG	-	9268 (0)	77.19
TAG	(1,1)	7988 (2)	**78.54**
	(100,1)	7462 (16)	75.44
	(100,100)	8057 (15)	77.98
Berkley Parser[8]	-	-	77.93[*1]
TSG[2]	-	-	78.40

表 2　少量データセットでの構文解析結果

	(b_1, b_2)	# rules (# aux. trees)	F1
CFG	-	35374 (-)	71.0
TSG	-	80026 (0)	85.0
TAG	(1,1)	64217 (4)	85.3
	(100,100)	65099 (25)	84.8
TSG[10]		-	82.6[*2]
TSG[2]		-	85.3

表 3　標準データセットでの構文解析結果

4　実験

提案手法を評価するために，構文解析で標準的に用いられている英語の WSJ Penn Treebank データを用

[*1] Results from [2].
[*2] Results on length $\leq$ 40.

(N$\bar{\text{P}}$ (N$\bar{\text{P}}$) (: –))
(N$\bar{\text{P}}$ (N$\bar{\text{P}}$) (ADVP (RB respectively)))
(P$\bar{\text{P}}$ (P$\bar{\text{P}}$) (, ,))
(V$\bar{\text{P}}$ (V$\bar{\text{P}}$) (RB then))
(V$\bar{\text{P}}$ (V$\bar{\text{P}}$) (RB not))
(Q$\bar{\text{P}}$ (Q$\bar{\text{P}}$) (IN of))
($\bar{\text{S}}$ ($\bar{\text{S}}$) (: ;))

表 4　獲得された補助木の例

いて実験を行った．実験設定は [3] と同じである．文法獲得の評価は，Treebank の学習データから TAG の基本木を教師なし学習で獲得した後，あらかじめ構文情報を除外したテストデータに対して構文解析を行い，精度（F1 スコア）を評価した．学習データは少量のデータセット（セクション 22：約 2000 文）と標準のデータセット（セクション 2-21：約 20000 文）を用いた．テストデータは，少量のデータセットではセクション 23 の全ての文を用い，標準のデータセットではセクション 24 の全ての文を用いた．

表 2 に少量データセットでの構文解析結果を示す．表中の TSG は，提案手法（TAG）の接合確率を $a_X = 0$ と設定したものである．(b_1, b_2) は，提案モデルのハイパーパラメータで，接合操作の起こりやすさを調節する値である．b_1 に大きな値を設定すると接合操作を行う確率を高め，b_2 に大きな値を設定すると接合操作を抑制する効果がある．

少量データでは，TSG と TAG は CFG よりも極めて高い精度で構文解析を実行できることがわかる．また，TSG と TAG を比較すると，TAG は適切なハイパーパラメータを設定すれば TSG よりも少し良い精度であった．したがって，少量のデータセットでは，接合操作が汎用性の高い基本木の獲得に貢献していると考えられる．また，他の構文解析手法と比較すると，現在標準的に使用されている Berkley Parser[8] や近年提案されたベイズモデルによる TSG 獲得手法 [2] よりも少し良い結果であった．したがって，本手法は構文木データがあまり入手できない状況では構文解析器として十分なものである．

表 3 に標準データセットでの構文解析結果を示す．少量データセットと同様に，TSG と TAG は CFG よりも極めて高い精度であったが，TSG と TAG ではほぼ同等の結果であった．これは，学習データが増加したことにより補助木の効果が減少したと考えられる．ただし，TAG は TSG と比較すると基本木の数が 20% 程度少ない．これは，TAG が接合操作によってコンパクトな基本木の集合だけで様々な構文木を生成できるためであると考えられる．また，TAG の初期木と補助木の数を比較すると，圧倒的に初期木の数が多く，補助木はわずかなバリエーションしか存在しなかった．ただし，本手法は基本木の初期状態を全て初期木としているため，学習がうまく進まずに適切な補助木を獲得できなかった可能性もある．また，英語以外の言語では傾向の異なる結果が得られる可能性もある．

表 4 に，標準データセットから獲得した補助木の例をいくつか示す．主に記号（“ – ”, “ , ”, “ ; ”）や副詞（RB）が接合操作によって挿入されやすいことがわかる．これは，英語では他の品詞と比べて記号や副詞は文の様々な位置へ挿入することができることからも直観的に理解可能であり，妥当な結果といえる．

5　まとめ

本稿では，構文木のコーパスから，コンパクトな TAG の基本木を自動的に獲得する方法を提案した．提案モデルは，基本木の確率分布に事前確率を導入したベイズモデルを用い，効率的な学習のために TAG のための基本木分解を考案した．本手法を用いて構文解析の実験を行ったところ，少量のデータでは TSG や他手法よりも少し良い結果が得られた．また，標準のデータでは TSG とほぼ同等の結果が得られるとともに，20% 程度コンパクトな基本木が獲得できることがわかった．本手法によって獲得された補助木は，記号や副詞など文の様々な位置に現れる単語を持つものが多いという妥当な結果を得た．

参考文献

[1] David Chiang. *Statistical Parsing with an Automatically Extracted Tree Adjoining Grammar*, chapter 16, pages 299–316. CSLI Publications, 2003.

[2] Trevor Cohn and Phil Blunsom. Blocked inference in Bayesian tree substitution grammars. In *Proc. of ACL*, pages 225–230, 2010.

[3] Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing compact but accurate tree-substitution grammars. In *Proc. of HLT-NAACL*, pages 548–556, 2009.

[4] Steve DeNeefe and Kevin Knight. Synchronous tree adjoining machine translation. In *Proc. of EMNLP*, pages 727–736, 2009.

[5] Mark Johnson, Thomas Griffiths, and Sharon Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proc. of HLT-NAACL*, pages 139–146, 2007.

[6] A.K. Joshi. Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions. *Natural Language Parsing: Psychological, Computational, and Theoretical Perspectives*, pages 206–250, 1985.

[7] Yudong Liu and Anoop Sarkar. Experimental evaluation of LTAG-based features for semantic role labeling. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 590–599, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[8] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proc. of ICCL-ACL*, 2006.

[9] J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2):855–900, 1997.

[10] Matt Post and Daniel Gildea. Bayesian learning of a tree substitution grammar. In *Proc. of ACL-IJCNLP*, pages 45–48, 2009.