構文的類似度を考慮した構文木コーパスの誤り訂正

鈴木 寛大 † 加藤 芳秀 ‡ 松原 茂樹 †
† 名古屋大学大学院情報科学研究科
‡ 名古屋大学情報連携統括本部
ksuzuki@db.ss.is.nagoya-u.ac.jp

1 はじめに

構文木コーパスは有用な言語資源として，自然言語処理の研究に活用されている．しかし，コーパスへのタグ付けの誤りの混入は避けがたく，これがコーパスの質を下げる問題となっている．これに対して，構文木コーパスに含まれる誤りを訂正する手法が提案されている [3, 5]．これらの手法は，誤った木構造を正しい木構造に変換するルールを作成し，コーパスに適用することで，誤り訂正を実現している．しかし，これらの手法が作成するルールの中には，同一の木構造を別の木構造に変換する複数のルールが含まれる場合がある．コーパス中の誤った木構造の出現それぞれに対し，適用すべきルールは異なるため，適用するルールを適切に選択する必要がある．

そこで本稿では，構文木コーパスに含まれる誤りに対して，複数の変換候補が存在する場合に，どのルールを適用すべきかを決定する手法を提案する．変換元，変換先それぞれの木構造の，コーパスでの出現における周辺の構造の構文的類似度を考慮して，いずれの木構造に変換することが適切かどうかを評価する尺度を定義する．Penn Treebank[4] を用いて実験を行ったところ，適切なルールの選択において，構文的類似度を考慮することの有用性が確認できた．

2 構文木コーパスの誤り訂正手法

本節では，構文木コーパスの誤り訂正手法である Kato らの手法 [3] と鈴木らの手法 [5] について説明し，その問題点を議論する．

これらの手法はいずれも synchronous tree substitution grammar (STSG)[2] に基づいて，誤り訂正を実現している．STSG は木構造を変換するルールから構成されるが，これを用いて，コーパス中の誤りを訂正する変換ルールを定義している．ルールの例を図 1 に示す．ルールは基本木と呼ばれる二つの木構造で構成される．両基本木の葉節点には，左から順に対応関係が取られる [1]．図 1 のルールを構文木に適用すると，構文木中の基本木 (1) にマッチする構造が，基本木 (2) に変換される．以下では，ルールを構成する基本木のうち，変換元の木をソース木，変換先の木をターゲット木と呼ぶ．従来の手法は，誤った木構造を正しい木構造に変換する誤り訂正ルールをコーパスに適用することで誤り訂正を行う．

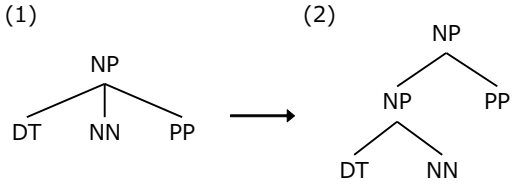

図 1: STSG ルールの例

Kato らの手法では，作成したルールが誤り訂正に適切かどうかを，基本木の出現頻度に基づいた尺度を用いて評価している．鈴木らの手法も，ルールの作成方法は異なるが，同一の尺度を用いている．この尺度は，ソース木 s をターゲット木 t に変換するルール $\langle s, t \rangle$ に対して以下のように定義される．

$$Score(\langle s, t \rangle) = \frac{f(t)}{f(s) + f(t)}$$

ここで，$f(\cdot)$ は，基本木のコーパス中の出現頻度を表す．ソース木の出現頻度に対して，ターゲット木の出現頻度が大きいほど Score は大きな値を取り，そのルールが誤り訂正に適切である可能性が高いと見なされる．

前述の手法においては，複数のルールが同一のソース木を持っている場合がある．同一のソース木を持つルールの例を図 2 に示す．ある構文木中に，このソース木が出現している場合，どちらのルールも適用可能であるが，正しく誤りを訂正できるルールは唯一であり，その他のルールを適用することは不適切である．さらに重要な点は，一方のルールが絶対的に正しい

[1] 一般的な STSG において，必ずしも左から順に対応がとられるとは限らない．

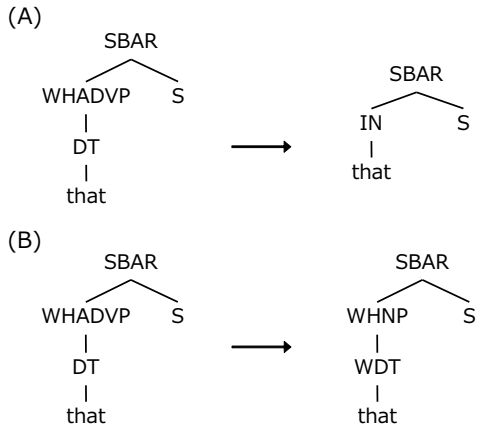

図 2: 同一のソース木を持つルールの例

わけではなく，適用箇所に応じて適切なルールも異なる点である．従来の手法では，Score の高いルールの適用が優先されるが，同一のソース木を持つルールの Score を比較すると，ターゲット木の出現頻度が最も高いルールの Score が最大になる．そのため，ソース木は，コーパス中のすべての出現箇所において，最も出現頻度が高いターゲット木に変換されてしまい，出現箇所ごとに異なるルールを適用することはできない．図 2 に示した 2 つのルールが適用可能な構文木の例を図 3 に示す． 図 2 の 2 つのルールのソース木にマッチする木構造は点線で囲まれている．構文木 (a) にはルール (A) を適用し，構文木 (b) にはルール (B) を適用しなければ正しく誤りを訂正できない．仮に，ルール (A) のターゲット木の方が出現頻度が大きいとすると，(a),(b) の両方にルール (A) が適用されるが，(b) に対するルール (A) の適用は不適切である．ルール (B) のターゲット木の出現頻度の方が大きいとすると，(a) に対して (B) が適用されてしまう．このような不適切なルールの適用を防ぐために，ソース木の出現箇所の情報を反映した評価尺度が必要である．

3 構文的類似度を考慮した誤り訂正ルールの選択

本節では，前節で述べた問題を解決するために，ソース木の出現箇所に応じて，適切な誤り訂正ルールを選択する手法を提案する．

提案手法の基本的な考え方を説明するために，図 3 の構文木 (b) に出現するソース木に対して，図 2 の 2 つのルールのどちらを適用するのが適切かを考える．適切なルールは (B) であるが，これはソース木やターゲット木の情報ではなく，ルールの適用箇所の周りの構造，具体的には，単語列 “ate an apple” を覆う節点

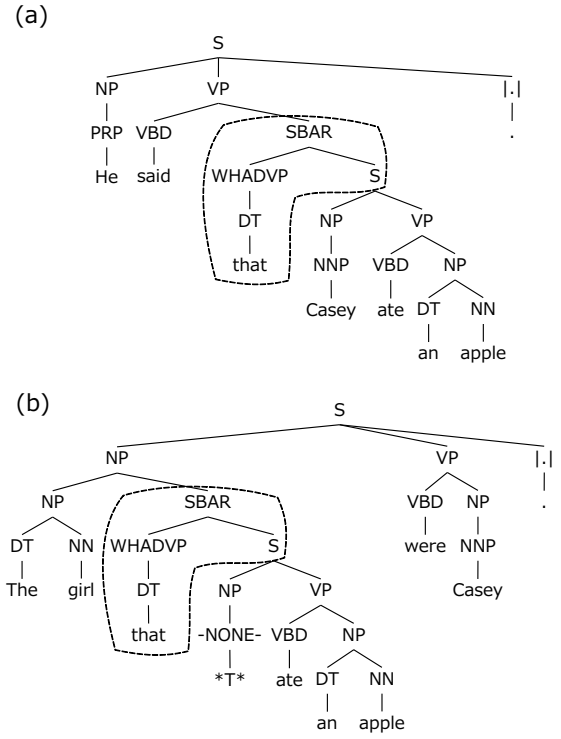

図 3: ルールが適用可能な構文木

S に wh 移動の痕跡*T*が含まれていることからわかる．このように，ソース木の周辺の構造に応じて適切なルールが定まる．

今，コーパスに出現するソース木の周辺の構造に注目したが，ここで，ターゲット木のコーパスの出現における周辺の構造に着目すると，例えば，図 4 の (c),(d) のような構文木が考えられる．構文木 (b) におけるソース木の出現の周辺の構造と，構文木 (d) におけるルール (B) のターゲット木の出現の周辺の構造には，ともに痕跡*T*を含むという類似性がある．

これを一般化すると，ソース木の出現における周辺の構造を参照し，各ターゲット木の出現が周辺に類似した構造を持つかを考慮することで，適切な変換を行うルールを選択できると期待できる．そこで，提案手法では，構文木の構文的類似度 [1] に基づき，構文木に出現する基本木の周辺の構造の類似度を定義し，これを用いて，ソース木の出現箇所ごとに誤り訂正ルールを評価する尺度を定義し，適用すべきターゲット木を決定する．

3.1 構文的類似度

本手法は，Collins ら [1] が定義する構文的類似度をベースとする．Collins らは，木構造の構文的類似度を，2 つの木構造が共通に含んでいる部分木の数と定義している．ここで言う部分木とは木構造グラフに含

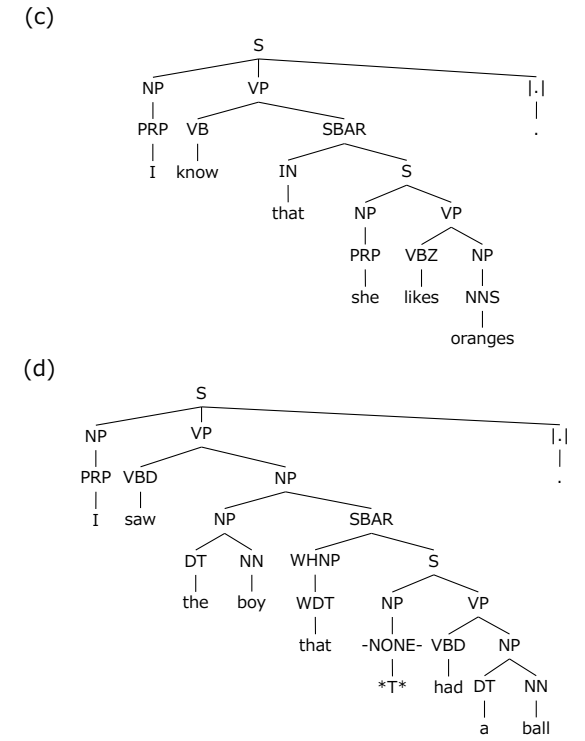

図 4: ルール (A),(B) のターゲット木を含む構文木

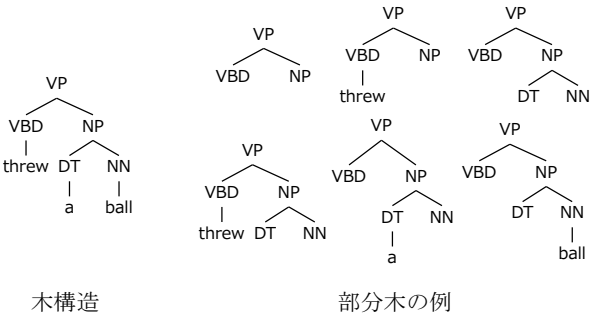

図 5: 構文木の部分木

まれる連結部分グラフのうち，以下の条件を満たすものを指す．

1. 2 つ以上の節点を持つ．
2. 木構造中の生成規則を保存している．

さらに，提案手法では次の条件を追加する．

3. 木構造の根を含む

図 5 に，木構造の部分木の例を示す．

次に，コーパス中の基本木の出現における周辺の構造の類似度について考える．提案手法では，出現する基本木の周辺の構造として，それが覆う木構造を参照する．すなわち，誤り訂正ルールの基本木の葉節点には対応がとられているが，コーパスに出現する基本木周辺の構造の類似度は，基本木の葉節点にマッチするコーパス中の節点が支配する木構造の類似度とする．コーパス中の構文木の節点 n_1 を根として出現する基本木 τ_1 と n_2 を根として出現する基本木 τ_2 の周辺の構造の類似度 $Sim(\tau_1, \tau_2; n_1, n_2)$ を以下の式で定義する．

$$Sim(\tau_1, \tau_2; n_1, n_2) = \sum_{i=0}^{nl(s)} C(L_{\tau_1}(n_1, i), L_{\tau_2}(n_2, i))$$

ここで，$C(x_1, x_2)$ は，節点 x_1, x_2 が支配する木構造の部分木のうち，共通する部分木の数である．$L_\tau(n, i)$ は，節点 n を根として出現する基本木 τ の左から i 番目の葉に対応する構文木中の節点である．また，$nl(\tau)$ は基本木 τ の葉節点の数である．

3.2 構文的類似度を考慮した評価尺度

コーパス中の構文木のある節点 n_s を根として出現するソース木 s は，その周辺の構造に類似した周辺を伴うターゲット木に変換すれば，適切に誤りを訂正できると期待できる．この考えに基づき，n_s を根として出現するソース木 s の，ターゲット木 t への変換の評価尺度 $Score(s, t; n_s)$ を以下のように定義する．

$$Score(s, t; n_s) = \sum_{n_t \in RO(t)} Sim(s, t; n_s, n_t)$$

ここで，$RO(t)$ は，コーパス中の節点の集合であり，ターゲット木 t がその要素を根として出現することを意味する．

出現するソース木ごとに，$Score(s, t, n_s)$ が最大になるルールを適用する．すなわち，節点 $n_s \in RO(s)$ を根として出現するソース木 s に対しては，$Score(s, t; n_s)$ が最大になるルール $\langle s, t \rangle$ を適用する．

誤り訂正手法 [3, 5] で作成されたルールには，正しい木構造を誤った木構造に変換するルールが含まれている場合があり，このような場合はそもそもルールを適用すべきではない．そこで，任意の基本木 τ について，ルール $\langle \tau, \tau \rangle$ を誤り訂正ルールに追加する．ただし，$Score(\tau, \tau; x)$ の計算において，$x \notin RO(\tau)$ とする．

4 実験

提案手法の有用性を確かめるために，構文木コーパスである Penn Treebank[4] を用いて実験を行った．

実験の概要は以下の通りである．Penn Treebank の Wall Street Jarnal コーパスの全 49208 文を対象に，鈴木らの手法を用いてルールを作成した．作成された全 2379 個のルールについて，同一のソース木を持つ

表 1: 選択ルール適用の判定結果

提案手法が選択したルールが適切	13
従来手法が選択したルールが適切	5
どちらのルールも不適切	21

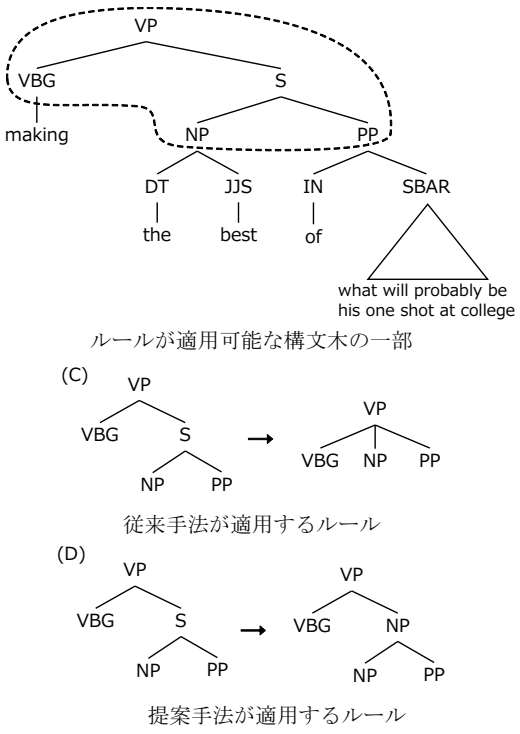

図 6: 適切なルールの選択に成功した例

ルールをひとつにまとめて集合とし，各ルールを従来手法の Score で評価し，ルール集合中の最も高い Score をキーにしてソートした．各ルール集合に対して，提案手法を用いて，ソース木の出現箇所ごとに適用するルールを選択した．いずれかの箇所で，従来手法と提案手法が異なるルールが適用されたルール集合上位 30 個に対し，異なるルールが適用された箇所へのそれぞれのルールの適用 (適用箇所の合計は 39 箇所である) が適切かどうかを判定した．選択したルールの適用が適切であるかどうかの内訳を表 1 に示す．提案手法により，適用箇所に応じてルールを適切に選択できるケースを確認できた．

従来手法とは異なるルールを選択し，適切なルールの選択に成功した例を図 6 に示す．構文木に出現するソース木の配下の構造のうち，節点 PP の配下の構造に注目すると，そこには [PP [IN “of”] [SBAR]] といったパターンが見られる．一般に，前置詞 of は，名詞のみを修飾し，動 (名) 詞を修飾しないが，(C) のターゲット木は動詞を修飾するパターンであり，その出現における PP の配下の構造に前置詞 of を含むことはなく，類似度が低くなり，結果として，提案手法の *Score* は低くなる．一方，(D) は名詞を修飾するパターンであり，PP の配下の構造として of を含む場合もあるため *Score* が高くなる．

5 おわりに

本稿では，コーパスの誤り訂正において，複数の誤り訂正ルールが適用できる場合に，適切なルールを選択する手法を提案した．提案手法では，構文的類似度を考慮することで，ソース木の出現箇所ごとに，各ルールの適用を評価する尺度を定義した．実験によって，適用ルールの選択において，構文的類似度を考慮することの有用性が確認できた．

本手法では，類似度のひとつとして，共通する部分木の数を用いたが，その他の類似度についても今後検討する予定である．

参考文献

[1] Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pp. 625–632. 2002.

[2] Jason Eisner. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 205–208, 2003.

[3] Yoshihide Kato and Shigeki Matsubara. Correcting syntactic annotation errors using a synchronous tree substitution grammar. *IEICE Transactions on Information and Systems*, Vol. E93-D, No. 9, pp. 2660–2663, 2010.

[4] 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.

[5] 鈴木寛大, 加藤芳秀, 松原茂樹. 半構造データマイニングに基づく構文木コーパスの誤り訂正. 言語処理学会第 21 回年次大会発表論文集, 2015.