局所的モデルと CKY アルゴリズムによる並列構造解析

寺西 裕紀[†]・進藤 裕之[†]・渡辺 太郎[†]・松本 裕治^{††}

並列構造解析とは等位接続詞によって結びつけられる句を同定するタスクである。並列構造は自然言語の曖昧性の主たる要因の一つであり、最高精度の構文解析器であっても誤りを生じさせる。本研究は、句のペアが並列となる場合に高いスコアを出力するようなスコア関数を定義し、解析時にはスコア関数と CKY アルゴリズムを組み合わせた構文解析によって並列構造を導出する。提案手法では並列構造解析を三つのサブタスクに分解し、それぞれのサブタスクを学習した三つのニューラルネットワークによってスコア関数を構成する。提案手法が並列構造を範囲の競合なく導出できることを保証しつつ、既存手法より高い精度で並列構造を同定できることを英語における評価実験により示す。

キーワード：並列構造解析, 並列句, CKY アルゴリズム, 構文解析

Coordinate Structure Analysis using Local Models and CKY Algorithm

HIROKI TERANISHI[†], HIROYUKI SHINDO[†], TARO WATANABE[†] and YUJI MATSUMOTO^{††}

The task of coordinate structure analysis is to identify coordinated phrases linked by a coordinator. Coordination is a major source of ambiguities in natural language and confuses even state-of-the-art syntactic parsers. In this paper, we generalize a scoring function that takes a pair of spans with a coordinator and returns a higher score for the two spans to be coordinated. During inference, our system employs this function with the CKY parsing algorithm and produces coordinate structures for a given sentence. To obtain such a function, we decompose the task into three independent subtasks and build the function based on three different neural networks for the tasks. The experimental results on English corpora demonstrate that our model achieves state-of-the-art results, ensuring that the global structure of coordination is consistent.

Key Words: *Coordinate Structure Analysis, Conjuncts, CKY Algorithm, Syntactic Parsing*

[†] 奈良先端科学技術大学院大学先端科学技術研究科, Graduate School of Science and Technology, Nara Institute of Science and Technology

^{††} 理化学研究所革新知能統合研究センター (AIP), RIKEN Center for Advanced Intelligence Project

1 はじめに

並列構造とは等位接続詞が結びつける句（並列句）から成る構造である．並列句の範囲の解釈には曖昧性があり，しばしば人間にとっても同定することが難しい．例えば，“*Toshiba’s line of portables, for example, features the T-1000, which is in the same weight class but is much slower and has less memory, and the T-1600, which also uses a 286 microprocessor, but which weighs almost twice as much and is three times the size.*”という文を一目見て，各等位接続詞に対する並列句を全てを見つけることは困難である．並列構造の存在は文を長くし，解釈を曖昧にするため，構文解析において誤りの要因となっている．

等位接続詞に対する並列句を同定する方法として，先行研究は並列句の二つの性質を利用してきた．(1) 類似性－並列句は類似した言語表現となる傾向がある．(2) 可換性－並列句を入れて替えても文全体が文法的に適格である．Ficler and Goldberg (2016b) は並列句ペアの類似性と可換性の特徴に基づいた計算を行うニューラルネットワークと構文解析器を組み合わせる方法を提案した．Teranishi, Shindo, and Matsumoto (2017) もこれらの二つの特徴を取り入れているが，構文解析の結果を用いずに最高精度の性能を達成している．どちらのアプローチも Shimbo and Hara (2007) や Hara, Shimbo, Okuma, and Matsumoto (2009) の類似性に基づく手法と比べて高い性能を得ているが，三つ以上の並列句を持つ並列構造や文中の複数の並列構造をうまく取り扱うことができない．特に文中に複数の並列構造が存在する場合には，並列構造の範囲が不整合に重なり合う状況が生じ得るという問題がある．対して，Hara et al. (2009) は並列構造の範囲に不整合が生じることなく並列構造を導出できる生成規則を用いている．

本論文では，並列構造解析における新しいフレームワークを提案する．このフレームワークでは等位接続詞と語系列上の二つの範囲（スパン）を取るスコア関数を用いる．スコア関数は二つのスパンが並列となる場合に高いスコアを返す働きを持つ．この関数を並列構造の導出規則に基づく CKY アルゴリズムと組み合わせることで，システムは入力文に対する並列構造の集合を範囲の競合なく出力する．このようなスコア関数を得るために，並列構造解析のタスクを等位接続詞の同定，並列句ペアの内側境界の同定，外側境界の同定の三つのサブタスクに分解し，それぞれに異なるニューラルネットワークを用いる．各ニューラルネットワークは並列構造の構成要素に対して局所的に学習を行うが，CKY アルゴリズムによる構文解析時に協調して働く．英語における評価実験の結果，我々のモデルは並列構造を範囲の競合なく導出できることを保証しつつ，Teranishi et al. (2017) の手法の拡張や先行研究と比較して高い精度を達成していることが示された．

本研究の貢献は以下のとおりである．

- 並列句ペアに対するスコア関数の学習・適用によって並列構造を解析するというフレームワークを提案した．

- 並列構造解析を三つのサブタスクに分解し, CKY アルゴリズムによる構文解析において協調して働くモデルを開発した.
- 三つ以上の並列句を含む並列構造や文中の複数の並列構造を範囲の競合なく導出可能なシステムを確立し, 既存手法を上回る解析精度を達成した.

2 並列構造解析

2.1 並列構造

並列構造 (coordinate structure) とは, 等位接続詞 (coordinator) が結びつける複数の句から成る統語構造である¹. 等位接続詞によって結び付けられた句は並列句 (conjunct) と呼ばれる. 英語において, and や or, but などの等位接続詞によって形成される並列構造に接続して, カンマ (,) やセミコロン (;) などとも等位接続の働きを持って並列句をさらに連結させる場合がある. 本論文では, このように等位接続詞と協調して等位接続の役割を果たす語を準等位接続詞と呼ぶ. 準等位接続詞は等位接続詞に付随せずに単体で並列構造を形成することができない. 並列構造は多くの場合, 等位接続詞の出現によって検出されるが, but が前置詞として働く場合があるように, 語の表層形のみから並列構造の有無を判定することはできない. 本論文では, 等位接続詞・準等位接続詞となり得る語をそれぞれ並列キー・準並列キーと呼ぶ. 英語においては as well as などの複数の語から成る表現が等位接続の働きをする場合があるが, 本研究では一語から成る等位接続詞を解析の対象とする.

2.2 タスク定義と困難さ

並列構造解析とは, 文に現れる並列キーに対して並列句を同定するタスクである. より具体的には, 並列キーが並列構造を形成する場合は, その並列構造に含まれる全ての並列句の範囲 (スパン) を [始点, 終点] の形式で同定する. 反対に並列キーが並列構造を形成しない場合は, 並列構造の不在を表す NONE を返す. 並列キーと見なされる語は解析に先立って定義されるものとする.

並列構造解析のタスクの難しさは複雑な並列構造の取扱いにくさから生じており, 代表的には二つのケースが挙げられる. 一つは並列構造が三つ以上の並列句を含む場合であり, もう一つは文中に複数の並列構造が存在する場合である. 並列構造は準等位接続詞を伴って三つ以上の並列句を持つ可能性があるため, カンマに代表される準並列キーが実際に等位接続の役割を果たしているかを判定し, どの並列構造に属して並列句を連結しているのかを突き止めなければならない. 文中に複数の並列構造が存在する場合, 各並列構造は別の並列構造に包含されて

¹ 並列句は句構造文法における句 (phrase) と必ずしも一致しないが, ここでは便宜上 “句” という表現を用いる.

入れ子の形で出現するか、もしくは他の並列構造と範囲が全く重なり合うことなく独立して出現する。言い換えると、二つの並列構造の範囲が部分的に重なり合うことはない。したがって、並列構造解析のタスクでは、各並列構造がいくつの並列句を含んでいるか知ることができない状態で、複数の並列構造の範囲の制約を満たしながら個々の並列句を同定しなければならない。

並列構造解析のタスクは句構造解析や依存構造解析などの構文解析によって必ずしも解くことができない点でも困難である。例えば、“I met [Bob on Sunday] and [Alice on Wednesday].” という文では、“Bob on Sunday” と “Alice on Wednesday” が並列となっているが、これらはふつう句構造文法の句と対応しない。また、“We saw an old [man] and [woman].” と “We saw a [beautiful woman] and [man].” という二つの文では、前者は old の主辞が man であり後者は beautiful の主辞は woman であり、どちらも同一の依存構造を持ち、Stanford/Universal Dependencies (de Marneffe, MacCartney, and Manning 2006; Nivre, de Marneffe, Ginter, Goldberg, Hajić, Manning, McDonald, Petrov, Pyysalo, Silveira, Tsarfaty, and Zeman 2016) のアノテーションでは二つの文の並列構造の範囲の違いを表すことができない。さらに、“[Mary always orders wine], and [Sally beer].” という文において、and に後続する並列句では “always orders” が省略されているように、空所化 (gapping) という現象が生じる場合がある。このような空所化を句構造や依存構造では適切に表現することができず、文の統語構造がしばしば不自然な構文木で表されるため、構文解析の障害になるという問題がある。したがって、句構造や依存構造などを対象にした構文解析の枠組みで並列構造を解析することは容易ではなく、並列構造解析に特化した手法の研究がなされている²。

2.3 並列構造の木構造による表現

並列構造の階層関係や並列構造に含まれる複数の並列句の関係は木構造として表現することができる。本論文では並列構造の木構造による表現を並列構造木と呼ぶ。図 1 は並列構造木の例である。木構造による並列構造の表現は、複数の並列構造の範囲が競合しない（部分的に重なり合う状態とならない）点、並列句の数を二つに限定せず制限なく表せる点で適している。言い換えれば、文に含まれる並列構造を木構造として出力することで、並列構造の範囲の制約は必然的に満たされる。

2.4 並列構造の範囲の組み合わせ

並列構造解析の一つの障害となる点として、並列構造に含まれる並列句の範囲の組み合わせ数が膨大になることが挙げられる。文の長さを N 、等位接統詞の出現位置を k とする。並列句の数を二つに限定し、並列句が等位接統詞に隣接すると仮定した場合、並列句範囲の組み合わせ数は

² 組合せ範疇文法など、複雑な並列構造を説明できる文法も存在するが、それらの文法における統計的構文解析によって必ずしも並列構造を精度良く解析できるわけではない。

e.g.) It also recommends better retirement and day-care benefits, and basing pay on education, experience and nurses' demanding work schedules.

図 1 並列構造木の例. coord は並列構造, conj は並列句, cc は等位接続詞, cc-sub は準等位接続詞を表す.

$(k-1) \times (N-k)$ 通りである. 等位接続詞が文のほぼ中心位置に出現するとき, その組み合わせ数はたかだか $(N/2)^2$ 通りである. しかし実際には並列句は必ずしも等位接続詞に隣接せず, “[A] and, on the other hand, [B]” のように等位接続詞と並列句の間に挿入句が出現する場合がある. このような場合を考慮すると並列句範囲の組み合わせ数は $k(k-1)/2 \times (N-k)(N-k+1)/2$ 通りとなり, 等位接続詞が文のほぼ中心位置に出現するとき, その組み合わせ数はたかだか $(N/2)^4/4$ 通りである. これらの組み合わせ全てについて句のペアがどの程度並列になっているかを計算する場合, 時間計算量 $O(N^4)$ を要し, 計算方法によっては膨大な時間がかかる.

実際には, 準等位接続詞の出現により三つ以上の並列句が出現する場合や文中に複数の並列構造が出現する場合を考慮すると, それらの組み合わせの総当たり数は爆発的に増加するため, 並列構造の範囲の組み合わせ全てについて範囲の妥当性を検証・比較することは現実的ではない. 本研究では局所的な評価・比較を帰納的に適用することにより, 計算量を抑えながら並列構造の範囲の組み合わせ全てについて考慮する.

3 並列句の特徴と従来研究

並列構造解析, すなわち等位接続詞が結びつける並列句の範囲同定において, 有用な二つの特徴がある (Ficler and Goldberg 2016b; Teranishi et al. 2017).

- 類似性: 同一の並列構造に属する並列句は, 句の構文構造・意味の点で類似する.
- 可換性: 同一の並列構造に属する並列句は, 互いに入れ替えても文の流暢性が保たれる.

(a) 構文構造・意味の点で類似する並列句の対.

The Tass news agency said the 1990 budget anticipates

[*expenditures of 489.9 billion rubles (US\$ 790.2 billion)*]_{NP} and [*income of 429.9 billion rubles (US\$ 693.4 billion)*]_{NP}.

(b) 並列句の対を互いに入れ替えた文.

図 2 並列句の特徴.

例として, “The Tass news agency said the 1990 budget anticipates income of 429.9 billion rubles (US\$ 693.4 billion) and expenditures of 489.9 billion rubles (US\$ 790.2 billion).” という文を挙げる. 等位接続詞 and に対する二つの並列句, “income of 429.9 billion rubles (US\$ 693.4 billion)” と “expenditures of 489.9 billion rubles (US\$ 790.2 billion)” はともに名詞句であり, ほぼ同一の品詞系列・句構造を持つ (類似性, 図 2a). また, これらの並列句を互いに入れ替えた場合であっても文法性を損なうことなく, 構文的に正しい文が成立する (可換性, 図 2b).

3.1 並列句の類似性に基づく従来手法

並列構造解析のタスクにおいて, これまで並列句の類似性に基づく手法が盛んに研究されてきた. まず並列句が共通して持つ品詞タグ・意味ラベル・形態情報・構文構造に基づいてスコアを計算することによって, 並列する名詞句ペアを同定する手法が開発された (Agarwal and Boggess 1992; Kurohashi and Nagao 1994; Resnik 1999; Hogan 2007). これらの研究に対し, Shimbo and Hara (2007) は並列句の類似度に基づく識別モデルを提案した. Shimbo らの手法において, 並列句の類似度は系列アラインメントのノードとエッジに割り当てられた統語・形態情報の特徴の重み付き和として計算され, そのパラメータは機械学習手法であるパーセプトロンによって調整される. Hara et al. (2009) は Shimbo らの手法を発展させ, 並列構造に特化した文脈自由文法規則と組み合わせ, 三つ以上の並列句を持つ並列構造や文中の複数の並列構造を範囲が競合することなく導出可能にした. Hanamoto, Matsuzaki, and Tsujii (2012) は Hara らの手法に

よる解析結果と主辞駆動句構造文法の解析器の出力を双対分解によって一致させることで精度を向上させた。

並列句の類似性に基づく並列構造の解析手法の発展により, 名詞句の並列などの類似度が大きい並列句に対しては解析精度が向上してきた。しかしながら, 動詞句や節の並列, 形容詞・動詞など異なる統語範疇を持つ句の並列など, 類似度が必ずしも高くない並列句の同定には課題が残った。

3.2 並列句の類似性に依存しない手法

非類似となる並列句をとらえる方法として, Kawahara and Kurohashi (2008) は類似性の素性を用いずに依存構造と格フレームに基づいて並列句の生成確率を学習し, 範囲同定を行う手法を開発した。Yoshimoto, Hara, Shimbo, and Matsumoto (2015) は Eisner アルゴリズム (Eisner 1996) の規則を拡張し, 依存構造とともに並列構造を同定する手法を提案している。

3.3 ニューラルネットワークを用いた並列構造解析

近年, 並列句の類似性・可換性に基づいたニューラルネットワークによるスコア計算を取り入れた並列構造解析の手法が研究されている。Ficler and Goldberg (2016b) は句の類似性のみならず可換性についても並列句ペアのスコア計算の素性として用いている。Ficler らの手法は三つのコンポーネントから成り立っている。まず始めに二値分類器を用いて等位接続の働きをし得る語が実際に並列構造を持つか否かを予測する。次に並列構造を持つと予測された等位接続詞に対して, Berkeley Parser (Petrov, Barrett, Thibaux, and Klein 2006) を用いて並列句ペアの候補を抽出する。最後にニューラルネットワークを用いて並列句ペアの候補に対してスコア付けをし, 最もスコアの高いペアをシステムの予測として出力する。Ficler らの手法は従来の非ニューラルネットワークによる手法と比較して高い精度で並列句の範囲を同定することができるが, 並列句ペアの候補抽出やスコア計算の過程において構文解析器の結果を用いており, 構文解析の結果に強く依存しているという欠点がある。パイプライン的なアプローチに対し, Teranishi et al. (2017) はニューラルネットワークによるエンドツーエンドの解析手法を開発した。Teranishi らの手法は直接的に個々の並列句を求めるのではなく, 並列構造全体の始点・終点を予測した後に, 並列構造を個々の並列句に分割するというトップダウンなアルゴリズムを採用している。並列句の類似性・可換性を考慮したニューラルネットワークによって並列構造の始点・終点が決定されるが, 構文解析の結果を用いないという点で Ficler らの手法と異なる。Teranishi らの手法は Ficler らの手法よりも高い解析精度を達成したが, Hara et al. (2009) の手法のように複数の並列構造を同時に考慮して, 範囲の競合なく並列構造を導出することができていない。

本研究は, (1) Hara らの手法のように並列構造の範囲が文全体として整合性を保つように制

約を与える, (2) Teranishi らの手法のように並列となる語系列のペアに対して高いスコアを割り当てられるモデルを導入する, という二点を組み合わせる試みとして位置づけられる. Hara らの手法は動的計画法によりボトムアップに並列構造を導出しているのに対して, Teranishi らの手法はトップダウンに並列句を同定しているという違いがある. また, 動的計画法の過程で並列句ペアの候補を比較する際に, 逐一ニューラルネットワークによってスコア計算を行うことは計算量・解析速度の点で現実的なアプローチではない. 本研究は並列句ペア候補のスコア計算を分割することで, ニューラルネットワークを用いながら動的計画法を適用可能にし, 並列構造を範囲の競合なく導出できるようにしたことで, 従来研究より優れた解析性能を達成している.

4 文脈自由文法規則を用いた構文解析

4.1 並列構造の構文解析

本節では並列構造を構文木として出力する方法について述べる³. 並列構造を包含する文の構文木は表 1 の文脈自由文法 (CFG) 規則から導出され, 並列構造木と一対一に対応付けすることができる. 並列構造を木として表現する利点については 2 節で述べたが, CFG 規則を用いる利点として, (1) 並列構造の範囲が競合するような候補を排除し, 並列構造木を導出できる組み合わせのみを探索できる, (2) CKY アルゴリズムの適用により効率的に並列構造木を導出できる, 点が挙げられる. 表 1 の CFG 規則は Hara et al. (2009) が開発した規則をもとに作られているが, 我々の規則が等位接続詞と並列句の間に任意の句が挿入された並列構造であっても導出できるのに対し, Hara らの規則は等位接続詞に隣接する並列句しか導出できないという違いがある. 本規則を用いることで, 並列構造のアノテーションが付与された Penn Treebank コーパス (Ficler and Goldberg 2016a) において, 99.5% の並列構造を構文木として導出することができる⁴.

表 1 の規則の適用により, 同じ文から異なる構文木が複数導出され得る. そこで構文木に対してスコアを割り当てることで, 最もスコアの高い構文木をシステムの出力とする.

$$\hat{T} = \arg \max_{T \in \mathcal{T}_G(s)} (\text{score}(T)) \quad (1)$$

ここで, $\mathcal{T}_G(s)$ は文脈自由文法 G によって文 s から導出可能な構文木の集合である.

³ 提案手法は任意の言語に適用できるが, 本節以降は英語を対象として手法の説明および実装・評価を行い, 英語以外の言語への適用については 6.3 節で議論する.

⁴ 導出不可能な並列構造は “[A] and [B] and [C]” のような等位接続詞の一つが準等位接続詞のように働く場合がほとんどであるが, このような表現であっても入れ子になった二つの並列構造として導出することができる.

表 1 並列構造木の導出規則

非終端記号	
COORD	Coordination; 並列構造
CONJ	Conjunct; 並列句
CC	Coordinating conjunction; 等位接続詞
CC-SUB	Sub-coordinator; 準等位接続詞
W	Word; 語
N	Non-coordination; 非並列構造
S	Sentence; 文
並列構造に関する規則	
(1) COORD	→ CONJ N? CC N? CONJ
(2) COORD	→ CONJ CC-SUB COORD
(3) CONJ	→ COORD
(4) CONJ	→ N
並列構造以外の規則	
(5) S	→ COORD
(6) S	→ N
(7) N	→ COORD N
(8) N	→ W COORD
(9) N	→ W N
(10) N	→ W
前終端記号の規則	
(11) CC	→ (and or but nor and/or)
(12) CC-SUB	→ (, ; :)
(13) W	→ *

(...|...) は括弧内のいずれかの語に, “*” は任意の語にマッチし, “?” は直前の要素の 0 または 1 回の出現を表す.

4.2 スコア計算と CKY アルゴリズムの適用

本研究では並列構造の構文木に対して, 非終端記号である COORD のノードと前終端記号のノードのみにスコアを割り当てる. COORD が付与されたノードに対するスコアは, 関数 $score_{coord}$ によって与えられる. 前終端記号について, その終端記号である語が並列キーとして定義された語の集合 S_{cc} または準並列キーとして定義された語の集合 S_{sub-cc} に属するとき, そのノードのスコアは関数 $score_{ckey}$ により計算される. 終端記号の語が S_{cc} または S_{sub-cc} のいずれにも属さない場合, 前終端記号は W に一意に決まり, そのノードのスコアは 0 とする. 以上を整理して, スコア関数 $score(T)$ は次のように表される.

$$score(T) = \sum_{\langle [i,j], \ell \rangle \in T} score_{node}([i, j], \ell) \quad (2)$$

$$score_{node}([i, j], \ell) = \begin{cases} score_{coord}(i, j) & (\ell = \text{COORD}) \\ score_{ckey}(i, \ell) & (\ell \in \{\text{CC}, \text{CC-SUB}, \text{W}\}, i = j, w_i \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub-cc}) \\ 0 & (\text{otherwise}) \end{cases} \quad (3)$$

ここで $\langle [i, j], \ell \rangle$ は構文木 T に含まれるノードであり、ノードのスパン $[i, j]$ とそのラベル ℓ を表している。また、 w_i は入力文の i 番目に出現する語を表す。並列構造のノード $\langle [i, j], \text{COORD} \rangle$ に対するスコア関数 $score_{coord}$ と、並列キーまたは準並列キーに該当する語の前終端ノードに対するスコア関数 $score_{ckey}$ の定義については次節にて述べる。

本研究では、CFG 規則の適用による並列構造の構文木の導出のために CKY アルゴリズムを用いる。CKY アルゴリズムを適用するために、CFG 規則はチョムスキー標準形に変換される⁵。CKY アルゴリズムを用いた CFG 構文解析では、動的計画法により式 (1) の結果 $\hat{T}$ を時間計算量 $\mathcal{O}(N^3)$ (N は入力文の単語数) によって効率良く求めることができる。

4.3 文の前処理・後処理法

アメリカ英語において、引用のあとに続くピリオドやカンマは引用符の内側に置かれる。引用符で囲まれた句が三つ以上並列する場合に、並列句の区切りとなる準等位接続詞が引用符内で閉じてしまう（例：“associations,” “societies” and “councils”）。このような並列構造は表 1 の規則 (2) から正しく導出することができず、規則 (2) の適用によって準等位接続詞直後の並列句が閉じ引用符から開始してしまう。そこで引用符の内側に置かれた句読点を取り扱う方法について示す。

引用符の削除・復元による方法

引用符の削除・復元による方法では、文の構文解析に先立って引用符を削除する。その後構文解析によって並列句の範囲を同定したあとに引用符を挿入する。例えば、“A,” “B” and “C,” という並列構造を持つ文に対して、システムが並列句の境界を $[A], [B]$ and $[C]$, と予測した場合、後処理によって “[A],” “[B]” and “[C],” と復元される。

引用符と句読点のスワップによる方法

引用符と句読点のスワップによる方法では、カンマと閉じ引用符がこの並び順で出現した場合に、構文解析に先立って位置を入れ替える。その際にスワップ位置を保持しておき、その後構文解析によって並列句の範囲を同定したあとに再びスワップすることで元の語順に戻す。例えば、“A,” “B” and “C,” という文に対して、システムが前処理後の並列句の境界を “[A]”, “[B]”

⁵ 右辺が三項以上の規則について、本研究では右から順にバイナリ規則に変換する。

and “[C]”, と予測した場合, 後処理によって “[A],” “[B]” and “[C],” と復元される.

本研究ではスワップによる前処理・後処理法を採用する⁶. 理由として, スワップによる方法では引用符を手がかりとして並列句の範囲を決定することができる点が挙げられる. また, イギリス英語のように閉じ引用符のあとにカンマが出現する場合, スワップによる方法では単に前処理・後処理が適用されずにそのままシステムによる予測ができるが, 削除・復元による方法ではそのような場合であっても引用符が削除されてしまい, システムの予測時に引用符を手がかりとして役立てることができない.

5 構文解析モデル

本節では 4 節で示した構文解析アルゴリズムに用いるスコア関数 $score_{coord}$, $score_{ckey}$ の定義と学習方法について示す. 提案手法ではスコア関数を, 等位接続詞分類モデル, 内側境界スコア付与モデル, 外側境界スコア付与モデルの三つのモデルにより構成する. 図 3 は本手法のフレームワークの概要図である.

5.1 モデル定義

N 語の系列 $w_{1:N} = w_1, \dots, w_N$ と対応する品詞タグ系列 $p_{1:N} = p_1, \dots, p_N$ を入力として, 我々のシステムは並列構造の集合 $\{\langle t, \{[b_t^{(1)}, e_t^{(1)}], \dots, [b_t^{(n)}, e_t^{(n)}]\} \rangle\} (n \geq 2, b_t^{(i)} \leq e_t^{(i)})$ を出力する. ここで t は等位接続詞の出現位置を表し, $[b_t^{(k)}, e_t^{(k)}]$ は等位接続詞 w_t によって形成される並列構造に含まれる並列句のうち先頭から k 番目に出現する並列句であり, その開始位置が $b_t^{(k)}$, 終了位置が $e_t^{(k)}$ であることを表す. ここで並列構造に含まれる並列句の数 n が $n \geq 2$ となっている理由は, 並列構造に含まれる並列句の数をあらかじめ知ることができないためである. しかし, 並列キーと準並列キーに対する並列句ペアの同定として, 以下のようにタスクを定式化することができる.

$$\begin{aligned} X &= \{w_{1:N}, p_{1:N}, C\} \\ C &= \{t \mid w_t \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub-cc}\} \\ Y &= \{\langle y_t^{ckey}, y_t^{pair} \rangle \mid t \in C\} \end{aligned} \quad (4)$$

ここで y_t^{ckey} は語 w_t が実際に等位接続の役割を果たしているか ($y_t^{ckey} = 1$), または果たしていないか ($y_t^{ckey} = 0$) を表す二値のラベルであり, y_t^{pair} は並列句スパンのペアである. また,

⁶ 前処理・後処理を加える方法の他に, 単に引用符を考慮した CFG 規則を追加することもできるが, CFG 規則を不用意に複雑にしてしまうため, 本研究では採用していない.

図 3 並列構造解析のフレームワークの概観. CKY アルゴリズムの適用において, 前終端記号の四角形ノードは等位接続詞分類モデルによって, COORD ノードは内側・外側境界スコア付与モデルによってそれぞれスコア付けされる.

$y_t^{key} = 0$ のとき, 対応する並列句ペアが存在しないことから $y_t^{pair} = \emptyset$ となる. $t = 1$ または $t = N$ の場合について, 語 w_t は文内に並列構造を形成することができないため $y_t^{key} = 0$ となる. 本論文では, 等位接続詞と成り得る語の集合 S_{cc} と準等位接続詞と成り得る語の集合 S_{sub-cc} をそれぞれ {“and”, “or”, “but”, “nor”, “and/or”} と {“,”, “;”, “.”} と定義する. 提案手法では, y_t^{pair} の四つの境界—(準) 等位接続詞の左側に出現する並列句の始点・終点, 右側の並列句の始点・終点—に対して, 内側境界 (左側並列句の終点・右側並列句の始点) を同定するモデル

と外側境界（左側並列句の始点・右側並列句の終点）を同定するモデルの二つを用いる⁷。四つの境界の組み合わせを列挙する場合の時間計算量は $O(N^4)$ であるが、内側・外側境界に分割して数え上げた場合は $O(N^2) + O(N^2) = O(N^2)$ となり、計算量を抑えることができる。

等位接続詞分類モデル

等位接続詞分類モデルは並列キーおよび準並列キーのラベルを予測する二値分類器である。

$$P_{\theta}(y_t^{ckey} | t) = \text{softmax}(f_{ckey}(t)) \quad (5)$$

ここで θ はモデルのパラメータ集合である。二値分類の損失関数は以下のとおりに計算される。

$$\ell_{\theta}^{ckey}(X, Y) = - \sum_{\langle y_t^{ckey}, y_t^{pair} \rangle \in Y} \log P_{\theta}(y_t^{ckey} | t) \quad (6)$$

内側境界スコア付与モデル

内側境界スコア付与モデルは並列句のペアに対し、その内側境界に基づいてスコアを付与するモデルである。本論文では、左側並列句の始点、終点、右側並列句の始点、終点をそれぞれ b^l, e^l, b^r, e^r と表す。並列キー w_t に対する内側境界 (e^l, b^r) のスコアは以下のように計算される。

$$\text{SCORE}_{\theta}^{inner}(e^l, b^r, t) = f_{inner}(e^l, b^r, t) \quad (7)$$

内側境界の確率値は全ての可能な内側境界の組み合わせにおけるスコアを正規化することで計算される。

$$I_t = \{(1, t+1), (1, t+2), \dots, (1, N), (2, t+1), \dots, (t-1, N)\} \quad (8)$$

$$P_{\theta}(y_t^{pair} = ([*, e^l], [b^r, *]) | t) = \frac{\exp(\text{SCORE}_{\theta}^{inner}(e^l, b^r, t))}{\sum_{(e^l, b^r) \in I_t} \exp(\text{SCORE}_{\theta}^{inner}(e^l, b^r, t))} \quad (9)$$

$$\ell_{\theta}^{inner}(X, Y) = - \sum_{\langle y_t^{ckey}, y_t^{pair} \rangle \in Y} y_t^{ckey} \log P_{\theta}(y_t^{pair} | t) \quad (10)$$

ここで項 $y_t^{ckey} \log P_{\theta}(y_t^{pair} | t)$ は、（準）並列キーが等位接続となる場合 ($y_t^{ckey} = 1$) のみ交差エントロピー損失が計算され、等位接続とならない場合 ($y_t^{ckey} = 0$) は 0 となることを意味する。

⁷ 四つの境界の分割方法として、「句のペアの“二つの始点”と“二つの終点”」や「“左側並列句の始点・終点”と“右側並列句の始点・終点”」のような分割も考えられる。しかし予備実験の結果、「左側並列句の始点・終点と右側並列句の始点・終点」による分割は、一方の並列句を決定するときに他方の並列句を考慮しないため、並列句の類似性や可換性のような並列句ペアの特徴を用いることができず、十分な解析性能が得られなかった。「句のペアの二つの始点と二つの終点」による分割は、並列句ペアの始点同士・終点同士の類似性や境界前後の文脈との整合性を考慮できるものの、左側並列句の終点と右側並列句の始点がほとんどの場合に等位接続詞と隣接することから、モデルの訓練時に左側並列句の始点と右側並列句の終点を独立に学習している状態に陥り、提案手法の分割と比較して高い精度が得られなかった。

外側境界スコア付与モデル

内側境界スコア付与モデルと同様に外側境界 b^l, e^r の確率 $P_\theta(y_t^{pair} = ([b^l, *], [*], e^r) | t)$ は外側境界の全ての可能な組み合わせ O_t から計算される. 同様に, 損失関数 ℓ_θ^{outer} はスコア関数 $\text{SCORE}_\theta^{outer}(b^l, e^r, t) = f_{outer}(b^l, e^r, t)$ を用いて定義される. ここで内側境界の集合 I_t と外側境界の集合 O_t は, 境界の可能な組み合わせが同じであるため等しい. 内側境界の確率 $P_\theta(y_t^{pair} = ([*, e^l], [b^r, *]) | t)$ と外側境界の確率 $P_\theta(y_t^{pair} = ([b^l, *], [*], e^r) | t)$ を用いて, 最も確率の高い並列句ペアは以下のように求められる.

$$\begin{aligned} y_t^{pair} = \arg \max_{(\hat{e}^l, \hat{b}^r)} & P_\theta([*, \hat{e}^l], [\hat{b}^r, *]) | t \\ & \cup \arg \max_{(\hat{b}^l, \hat{e}^r)} P_\theta([\hat{b}^l, *], [*], \hat{e}^r) | t \end{aligned} \quad (11)$$

ただし, 式 (11) によって個々の (準) 等位接続詞について並列句ペアを決定すると範囲に不整合が生じる可能性があるため, 並列構造の解析時には4節で示したとおり, CFG 規則によって導出される構文木のスコアが最大となるように並列構造が決定される.

5.2 スコア関数の構成

本小節では4節で示したスコア関数 $score_{coord}$, $score_{ckey}$ の具体的な定義について説明する. 並列キーまたは準並列キーに該当する語 $w_i \in \mathbb{S}_{cc} \cup \mathbb{S}_{sub-cc}$ の前終端ノードに対するスコアは, (準) 並列キーの有無を表す二値のラベル y_i^{ckey} の対数確率から割り当てられる.

$$score_{ckey}(i, \ell) = \begin{cases} \log P(y_i^{ckey} = 1 | i) & (\ell \in \{\text{CC}, \text{CC-SUB}\}) \\ \log P(y_i^{ckey} = 0 | i) & (\ell = \text{W}) \end{cases} \quad (12)$$

並列構造を表す COORD ノードのスコアは (準) 等位接続詞が結びつける二つの並列句に基づいて計算される. 等位接続詞の出現位置を k , 左側並列句のスパンを $[i, l]$, 右側並列句のスパンを $[m, j]$ とすると, 語 w_k によって結びつけられた二つの並列句から成る並列構造 $[i, j]$ のスコアは, 内側境界 (l, m) の対数確率と外側境界 (i, j) の対数確率の和として計算される.

$$\begin{aligned} score_{coord}(i, j) &= \log P(y_k^{pair} = ([i, l], [m, j]) | k) \\ &= \log P([*, l], [m, *]) | k + \log P([i, *], [*], j) | k \end{aligned} \quad (13)$$

式 (13) の計算と表 1 の規則 (1) を対応づけると, 規則 (1) は以下のように表される.

$$\text{COORD}_{i,j} \rightarrow \text{CONJ}_{i,l} \text{ N}_{l+1,k-1} ? \text{CC}_{k,k} \text{ N}_{k+1,m-1} ? \text{CONJ}_{m,j}$$

準等位接続詞によって並列句が結びつけられる場合は, 準等位接続詞 w_k が結びつける左側並列句 $[i, l]$ と右側並列句 $[m, j]$ を用いて, 式 (13) と同様の計算を適用することで COORD ノード

のスコアが割り当てられる. 式 (13) の計算を表 1 の規則 (2) に当てはめた場合, 規則 (2) の右辺を展開して以下のように表される.

$$\text{COORD}_{i,j} \rightarrow \text{CONJ}_{i,l} \text{ CC-SUB}_{k,k} \text{ CONJ}_{m,j} \dots \text{N? CC N? CONJ}$$

(ただし, $l = k - 1, m = k + 1$)

5.3 ニューラルネットワーク

本小節では関数 f_{ckey} , f_{inner} , f_{outer} のニューラルネットワークによる構成について説明する.

エンコーダ

単語・品詞タグの系列の文レベルのベクトル表現を得るために, 本研究では双方向型 Long Short-Term Memory (BiLSTMs) (Hochreiter and Schmidhuber 1997) を用いる.

$$\mathbf{h}_{1:N} = \text{BiLSTMs}(f_{input}(w_{1:N}, p_{1:N})) \quad (14)$$

ここで LSTM の隠れ状態ベクトルの次元を d^{hidden} とすると, エンコーダから出力される系列の各ベクトル $\mathbf{h}_t$ の次元は $2d^{hidden}$ となる. BiLSTMs の入力として, 単語・品詞タグを各ベクトル表現にマッピングする関数 f_{input} が出力するベクトル系列を用いる. f_{input} 内での具体的なマッピング方法として, 事前学習したベクトル表現のほか, ELMo (Peters, Neumann, Iyyer, Gardner, Clark, Lee, and Zettlemoyer 2018) や BERT (Devlin, Chang, Lee, and Toutanova 2019) などの文脈を考慮した単語分散表現, 文字レベルでの演算を行う LSTM や畳み込みニューラルネットワークの出力ベクトルなどを用いることができる. これらの選択による性能差は 6 節で示す. f_{input} と BiLSTMs から成るモデルを本論文ではエンコーダと呼び, エンコーダは等位接続詞分類モデル, 内側境界スコア付与モデル, 外側境界スコア付与モデルの三つのネットワークの共通の下層部分として共有される.

等位接続詞分類モデル

f_{ckey} には並列キーの文レベルベクトル表現の線形変換を用いる.

$$f_{ckey}(t) = \mathbf{W}^{ckey} \mathbf{h}_t + \mathbf{b}^{ckey} \quad (15)$$

ここで $\mathbf{W}^{ckey} \in \mathbb{R}^{2 \times 2d^{hidden}}$ と $\mathbf{b}^{ckey} \in \mathbb{R}^2$ は等位接続詞分類モデルのパラメータである.

内側境界スコア付与モデル

エンコーダが出力した文レベルベクトル表現を用いて, 内側境界スコア付与モデルは内側境界に位置する二つのベクトル表現を連結し, 多層パーセプトロン (MLP) に入力することでス

カラー値を出力する.

$$f_{inner}(e^l, b^r, t) = \mathbf{w}_2^{in} \text{ReLU}(\mathbf{W}_1^{in}[\mathbf{h}_{e^l}; \mathbf{h}_{b^r}] + \mathbf{b}_1^{in}) + \mathbf{b}_2^{in} \quad (16)$$

ここで $\mathbf{W}_1^{in} \in \mathbb{R}^{d^{in} \times 4d^{hidden}}$, $\mathbf{b}_1^{in} \in \mathbb{R}^{d^{in}}$, $\mathbf{w}_2^{in} \in \mathbb{R}^{d^{in}}$, $\mathbf{b}_2^{in} \in \mathbb{R}$ は内側境界スコア付与モデルのパラメータである.

外側境界スコア付与モデル

外側境界スコア付与モデルは外側境界のベクトル表現と並列キー前後のベクトル表現の差を用いてスコア計算を行う. 減算は二つのスパンの意味的な距離や関連性をとらえる意図で導入している (Teranishi et al. 2017). 演算によって得られたベクトル表現を MLP に入力することでスカラー値を得る.

$$f_{feature}(b^l, e^r, t, \mathbf{h}_{1:N}) = [\mathbf{h}_{b^l} - \mathbf{h}_{t+1}; \mathbf{h}_{e^r} - \mathbf{h}_{t-1}] \quad (17)$$

$$f_{outer}(b^l, e^r, t) = \mathbf{w}_2^{out} \text{ReLU}(\mathbf{W}_1^{out} \mathbf{r} + \mathbf{b}_1^{out}) + \mathbf{b}_2^{out} \quad (18)$$

$$\mathbf{r} = f_{feature}(b^l, e^r, t, \mathbf{h}_{1:N})$$

ここで $\mathbf{W}_1^{out} \in \mathbb{R}^{d^{out} \times 4d^{hidden}}$, $\mathbf{b}_1^{out} \in \mathbb{R}^{d^{out}}$, $\mathbf{w}_2^{out} \in \mathbb{R}^{d^{out}}$, $\mathbf{b}_2^{out} \in \mathbb{R}$ は外側境界スコア付与モデルのパラメータである.

5.4 学習

ニューラルネットワークモデルのパラメータ集合 θ は以下の損失関数を最小化することによって最適化される.

$$L(\theta) = \sum_{(X,Y) \in D} (\ell_{\theta}^{key}(X,Y) + \ell_{\theta}^{inner}(X,Y) + \ell_{\theta}^{outer}(X,Y)) \quad (19)$$

ここで D は学習データに含まれる文 X とその文に含まれる並列構造 Y の対の集合である. 式 (19) で計算される損失は三つのサブモデルの損失の和であり, したがって三つのサブモデルのパラメータは同時に学習される.

なぜ局所的な学習を行うか

各サブモデルの局所的な決定に基づいてスコア関数を学習する代わりに, モデルを統合して CKY アルゴリズムによって導出された構文木のスコアと正解の構文木のスコアを最大マージン法などによって直接的に学習することができる (Stern, Andreas, and Klein 2017). しかしながら, このような大域最適の学習には非常に多くの時間を要し, またハイパーパラメータの注意深い設定が必要なうえ, 予備実験の範囲では Teranishi et al. (2017) の手法をわずかに上回る程

度の精度しか得られなかった。

内側・外側境界の局所的な学習の成功は、交差エントロピー損失を用いて学習を行ったことが要因であると分析する。最大マージン損失を最小化する大域的な学習の場合、スコア最大の構文本と正解の構文本に含まれる並列句の境界についてモデルのパラメータが調整される。対して交差エントロピー損失を用いた内側・外側境界の局所的な学習の場合、正解の境界だけでなく全境界についてパラメータの調整が行われ、不正解の境界全てのスコアが押し下げられるよう学習が進む。そこで交差エントロピー損失最小化による局所的な学習と類似の学習方法として、入力文 s に対する構文本 $T \in \mathcal{T}_G(s)$ の条件付き確率 $P(T|s)$ を次式のようにモデル化することを考える。

$$P_\theta(T|s) = \frac{\exp(\text{score}_\theta(T))}{Z_\theta(s)}, \quad Z_\theta(s) = \sum_{T' \in \mathcal{T}_G(s)} \exp(\text{score}_\theta(T')) \quad (20)$$

この条件付き確率に対して最尤推定によってパラメータ集合 θ を調整することで、局所的な学習による方法と同等以上の解析精度が期待できるが、分配関数 Z において文脈自由文法 G によって文 s から導出可能な構文本の全てについてスコア計算をする必要があり、現実的には計算が困難である。そこで分配関数 Z を何らかの方法により近似することで、このような大域的な学習を実行する方法も考えられる (Andor, Alberti, Weiss, Severyn, Presta, Ganchev, Petrov, and Collins 2016)。

しかし近年、Teng and Zhang (2018) や Dozat and Manning (2017) の手法に見られるように、構文解析のタスクにおいて局所的な決定をニューラルネットワークによって高精度に学習し、解析時に大域最適解を求めることで、大域的な目的関数によって学習したモデルの解析結果と遜色ない精度が実現できている。提案手法において並列句ペアの内側・外側境界は局所的に学習されるが、並列構造の同定時には CKY アルゴリズムを用いた構文解析によって両境界の整合性を考慮して大域最適解を求めている。このことから、各サブタスクを高い精度で学習できれば大域的な目的関数を用いた学習との解析精度の差は縮まると推測され、本研究でも局所的な学習による方法を採用する。

6 実験

6.1 実験設定

データセット

本研究では並列構造のアノテーションが付与された Penn Treebank (Ficler and Goldberg 2016a) (PTB) と GENIA Treebank beta (Kim, Ohta, Tateisi, and Tsujii 2003) (GENIA) を用いて評価実験を行う。実験の前処理・後処理には 4.3 節で示したスワップによる方法を用いる。PTB で

の実験では、コーパスの Wall Street Journal のパートのうち、セクション 2 から 21 を訓練データ、セクション 22 を開発データ、セクション 23 をテストデータとして用いる。GENIA での実験では、Hara et al. (2009) と同様に 5 分割交差検定によって評価を行う。

モデル

エンコーダ内の関数 f_{input} として、単語を事前学習済みの単語ベクトルに、単語を構成する文字列を文字ベースの畳み込みニューラルネットワーク (Ma and Hovy 2016) (CharCNNs) の出力ベクトルに、品詞タグを品詞タグの表現ベクトルにマッピングし、それらの三つのベクトル表現を連結したベクトルを出力する関数を用い、この設定を本論文では *default* と呼ぶ。事前学習済みの単語ベクトルとして、PTB での実験には GloVe (Pennington, Socher, and Manning 2014) を、GENIA での実験には BioASQ (Tsatsaronis, Schroeder, Paliouras, Almirantis, Androutsopoulos, Gaussier, Gallinari, Artieres, Alvers, Zschunke, and Ngomo 2012) を初期値として用いる⁸。PTB での実験において、品詞タグは Stanford POS Tagger (Toutanova, Klein, Manning, and Singer 2003) を用いて 10 分割ジャックナイフ法により付与し、GENIA での実験では先行研究 (Hara et al. 2009; Fidler and Goldberg 2016b; Teranishi et al. 2017) に従って、コーパスに付与されている品詞を用いた。モデルのパラメータの最適化は Adam (Kingma and Ba 2015) を用いて確率的勾配降下法によって行った。その他のハイパーパラメータについては表 2 のとおりである。

ベースライン

評価実験のベースラインとして Teranishi et al. (2017) の手法を再実装したものをを用いる。Teranishi らの手法は、並列構造全体の始点・終点を予測したあとに、等位接続詞とカンマによって並列構造を個々の並列句に分割している。しかし個々の並列構造に対して独立に始点・終点を決めているため、文中の他の並列構造と範囲が競合し得る。そこで Teranishi らの手法を拡張し、複数の並列構造の範囲が部分的に重なり合わないという制約のなかで、並列構造の範囲のスコアの合計が最も高くなる組み合わせを探索して決定する。ベースラインのモデルで使われるエンコーダについては、上述の *default* の設定と同様の構成を用いる。本論文では以降、この拡張を施したベースラインモデルを *Teranishi+17:+ext* として参照する。また PTB での実験では、Stern et al. (2017) の句構造構文解析器を用いて、並列構造の構文木表現を直接学習・予測した結果とも比較する。

評価

PTB での実験では、並列キーに対する並列句スパンの予測に基づいて、適合率 (P)、再現率

⁸ それぞれ <http://nlp.stanford.edu/data/glove.6B.zip> (glove.6B.100d.txt), [http://bioasq.lip6.fr/tools/BioASQword2vec/\(vectors.txt\)](http://bioasq.lip6.fr/tools/BioASQword2vec/(vectors.txt)) を使用した。

表 2 実験に用いたハイパーパラメータの設定.

項目名	値
単語分散表現の次元数 (GloVe)	100
単語分散表現の次元数 (BioASQ)	200
品詞タグのベクトル表現の次元数	50
CharCNNs 内の文字ベクトル表現の次元数	10
CharCNNs の畳み込みウィンドウ幅	5
CharCNNs の出力ベクトルの次元数	50
LSTM の隠れ状態ベクトルの次元数 d^{hidden}	512
BiLSTMs のレイヤー数	2
MLP の中間層のユニット数 d^{in}	1024
MLP の中間層のユニット数 d^{out}	1024
Dropout の適用確率 (all)	0.50
学習率の初期値	0.001
L2 正則化の強度 λ (PTB)	0.0
L2 正則化の強度 λ (GENIA)	0.0001
勾配クリッピングの閾値	5.0

(R), それらの調和平均である F1 値 (F) によってシステムの評価を行う. Teranishi et al. (2017) の評価実験に則って, 並列句の一致について以下四つの基準で評価する⁹.

- **whole**: 最初の並列句の始点と最後の並列句の終点の一致.
- **outer**: 最初の並列句の始点・終点の一致かつ最後の並列句の始点・終点の一致.
- **inner**: 等位接続詞前後の並列句それぞれの始点・終点の一致.
- **exact**: 全ての並列句の始点・終点の一致.

また, 評価に際して特に名詞句の並列構造のみを対象にした解析性能についても調べる¹⁰.

GENIA での実験では, 上述の一致基準に基づいて再現率によって評価する. ただし, 先行研究では whole の一致基準についてのみ評価を行っている. また, 並列構造の統語範疇ごとに対象を絞って性能を評価する.

先行研究との実験設定の違い

提案手法と先行研究の実験設定の違いについて整理する. 各手法が利用している外部資源について表 3 に示す. 提案手法, ベースラインおよび Teranishi et al. (2017) のモデルは, PTB と GENIA のいずれにおいてもコーパス外のテキストで事前に学習した単語ベクトル表現を用い

⁹ 並列構造が二つの並列句から成る場合は inner と outer の一致基準は同一である.

¹⁰ Fidler and Goldberg (2016b) と同様に, NP に加えて NX の並列構造も名詞句の並列構造と見なす.

表 3 比較手法で用いられる外部資源

(a) PTB での設定

	品詞タグ	句構造構文木	単語ベクトル	品詞ベクトル
Ours	Stanford POS Tagger	なし	GloVe*	ランダム
Teranishi+17:+ext	Stanford POS Tagger	なし	GloVe*	ランダム
Teranishi+17	Stanford POS Tagger	なし	Word2Vec*	ランダム
Stern+17	Stanford POS Tagger	なし	ランダム	ランダム
Ficler+16	Berkeley Parser	Berkeley Parser	Word2Vec	Word2Vec

(b) GENIA での設定

	品詞タグ	句構造構文木	単語ベクトル	品詞ベクトル
Ours	コーパス付随	なし	Word2Vec*	ランダム
Teranishi+17:+ext	コーパス付随	なし	Word2Vec*	ランダム
Teranishi+17	コーパス付随	なし	Word2Vec*	ランダム
Ficler+16	コーパス付随	Berkeley Parser	Word2Vec	Word2Vec
Hara+09	コーパス付随	なし	なし	なし

*はコーパス外のテキストを用いて学習したものを表す.

ているのに対し, Stern et al. (2017) の構文解析器ではランダムに初期化した単語ベクトルを, Ficler and Goldberg (2016b) のモデルは PTB・GENIA のそれぞれで事前学習したベクトル表現を用いている. PTB での評価実験において, Ficler らは Berkeley Parser によって付与された品詞タグを用いているのに対し, その他のモデルは Stanford POS Tagger で付与した品詞タグを用いている. GENIA においてはいずれもコーパスに付与されている品詞タグを用いている. また, Ficler らは PTB・GENIA のいずれにおいても Berkeley Parser が付与した句構造の構文木を用いており, 品詞タグのベクトル表現については Word2Vec (Mikolov, Chen, Corrado, and Dean 2013) を用いてそれぞれのコーパスで事前学習している. 外部資源の利用における差異は並列構造の解析性能に影響を及ぼすため, 本節の評価実験は厳密に同条件下での比較ではない¹¹. しかしながら, 本節では提案手法における外部資源の利用の有無による性能差を提示することで, 設定の違いを考慮したうえでの優位点を示す.

外部資源の利用以外の差異として, 4.3 節に示した引用符によるイレギュラーな並列構造に対する取扱いが挙げられる. Ficler らの手法では Berkeley Parser を用いて並列句ペアの候補を抽出しており, イレギュラーな並列構造を同定できるかどうかは構文解析の結果に依存する. ベースラインおよび Teranishi らの手法は, 引用符による変則的な並列構造を導出できないという制

¹¹ Ficler らは Berkeley Parser を改変して用いており, 改変部分や手法の実装についての詳細は得られなかった.

限はない. 対して提案手法は表 1 で示した文脈自由文法規則からイレギュラーな並列構造を導出できないため, 4.3 節で提案したスワップによる前・後処理と組み合わせることで対応する. 表 1 の規則によって変換された正解の構文木を Stern らの構文解析器によって直接的に学習する方法においても, 同様の理由で学習用の構文木を構築する前に前処理を行い, 解析器による構文木の出力後に後処理を行って評価をしている. このように, スワップによる前・後処理は表 1 の規則を引用符によるイレギュラーな並列構造に対応させる目的で導入されており, 提案手法のシステムの一部に組み込まれたものである. したがって, タスク評価の入出力時点では提案手法と他の手法で単語系列や並列構造範囲に差異はない. しかしながら, スワップによって単語系列が操作されることで引用符とカンマの語順が正規化され, BiLSTMs を用いた本手法にとってより有効な語順に変化している可能性がある.

最後に本論文で提案する文脈自由文法規則と Hara et al. (2009) の規則の違いによる性能差について述べる. Hara らの規則は等位接続詞の前後の並列句が等位接続詞に隣接すると仮定しており, “[A] ... and [B]” のような, 等位接続詞に隣接しない並列句を導出することができない. ただし, Hara らは “[A], and [B]” のような, 等位接続詞とその前に出現する並列句とのあいだにカンマが出現することを許容し, 規則を追加している. 対して我々は表 1 の規則 (1) によって, 等位接続詞と並列句のあいだに任意の句が出現するような並列構造を導出可能にしている. このような並列構造は PTB に実際に存在するが, GENIA においては等位接続詞と並列句のあいだにカンマ以外の任意の句が出現せず, アノテーションされた全ての並列構造が Hara らの規則により導出可能である. したがって GENIA での評価実験において, 規則の違いによる性能差は生じない¹².

6.2 実験結果

PTB と GENIA の実験結果をそれぞれ表 4 と表 5 に示す. PTB において, 提案手法 (Ours) は並列構造の全ての一致基準において既存手法の精度を大幅に上回った. 提案手法が並列句ペアの内側・外側境界について学習しているのに対し, ベースラインは並列構造全体の境界のみを学習しているため, whole 以外の基準では提案手法がより正確に境界を予測できている. 次に Stern et al. (2017) の句構造構文解析器による解析では, いずれの一致基準についても並列構造に特化した解析器と比べて並列構造の同定精度が低いことが確認された. しかし並列構造の構文木表現の構文解析精度について評価したところ, 句構造のラベル付きスパンの一致についての F1 値が開発データで 97.88%, テストデータで 97.63% という結果となった. この精度の高さは, 表 1 の CFG 規則で導出される構文木の大部分が W と N のノードによって右辺に展開さ

¹² 我々の構文解析器において Hara らの規則を用いることで, 正解の並列構造を候補から落とすことなく並列句の探索範囲を絞り込める点で, 提案規則を用いる場合と比べて優位な設定で評価実験を行えるが, 本節では表 1 の提案規則を用いて実験を行う.

表 4 PTB での実験結果

(単位:%)

		全並列構造						名詞句 (NP) 並列					
		開発			テスト			開発			テスト		
		P	R	F	P	R	F	P	R	F	P	R	F
Ours	whole	78.73	79.00	78.86	77.27	77.34	77.30	79.81	79.63	79.72	79.71	79.45	79.58
	outer	77.55	77.83	77.69	75.63	75.70	75.67	79.35	79.17	79.26	78.75	78.50	78.62
	inner	79.20	79.48	79.34	77.45	77.52	77.48	80.27	80.09	80.18	80.03	79.77	79.90
	exact	77.32	77.59	77.45	75.45	75.52	75.48	78.89	78.71	78.80	78.43	78.18	78.30
Teranishi+17 :+ext	whole	78.78	77.94	78.36	77.36	76.52	76.94	78.52	77.80	78.16	78.72	78.34	78.53
	outer	74.49	73.70	74.09	72.03	71.24	71.63	76.67	75.97	76.32	75.36	75.00	75.17
	inner	76.04	75.23	75.63	74.14	73.33	73.74	77.82	77.11	77.47	77.44	77.07	77.25
	exact	74.13	73.34	73.74	71.48	70.70	71.08	76.21	75.51	75.86	75.20	74.84	75.01
Teranishi+17	whole	75.92	72.87	74.36	—	—	—	77.90	75.05	76.45	—	—	—
	outer	72.48	69.57	70.99	—	—	—	76.24	73.45	74.82	—	—	—
	inner	74.07	71.10	72.56	73.46	72.16	72.81	77.43	74.59	75.99	75.87	74.76	75.31
	exact	72.11	69.22	70.63	—	—	—	75.77	72.99	74.35	—	—	—
Ficler+16	inner	72.34	72.25	72.29	72.81	72.61	72.7	75.17	74.82	74.99	76.91	75.31	76.1
Stern+17	whole	77.61	71.51	74.44	74.54	66.97	70.55	79.80	74.65	77.14	76.97	71.68	74.23
	outer	75.67	69.72	72.58	72.91	65.50	69.01	78.32	73.27	75.71	76.11	70.88	73.40
	inner	77.61	71.51	74.44	73.82	66.33	69.87	80.54	75.34	77.85	76.46	71.20	73.73
	exact	75.29	69.36	72.20	71.79	64.50	67.95	77.58	72.58	75.00	74.39	69.28	71.74

Teranishi+17 および Ficler+16 の数値は論文で報告されている結果より抜粋.

れる二分木であるためだと考えられる. しかし並列構造について高い精度で同定できていないことから, W や N のノードと比較して出現数が少ない COORD のノードについては句構造構文解析によって高い精度でスパンを予測できていないと推測される.

GENIA において, 提案手法は exact の基準でベースラインを上回る精度を達成した. 我々のモデルはベースラインと比較して, 並列構造全体の始点・終点を予測する能力について劣っているものの, inner の基準では高い性能を発揮している. 対して, ベースラインは whole の基準で高い性能を示しており, 他の基準では精度が相対的に低い. この性能差は二つの手法のアルゴリズムの違いを反映している. 提案手法はボトムアップに並列句ペアから並列構造を導出しているのに対し, ベースラインは並列構造全体の範囲を予測してから個々の並列句に分割するというトップダウンの解析を行っている. そのためベースラインの手法では, 文や節などの長い並列構造を非準等位接続詞であるカンマによって誤って分割し, 解析精度が低下してしまう. 提案手法の欠点は CKY アルゴリズムの初期段階の誤りが後段のステップまで伝搬することであり, そのため inner 以外の基準では誤りの影響を受けて精度が低下していると考えられる.

表 5 GENIA での実験結果

(再現率, 単位:%)											
		NP	VP	ADJP	S	PP	UCP	SBAR	ADVP	Others	All
	#	2317	465	321	188	167	60	56	21	3	3598
Ours	whole	70.04	66.88	85.98	54.25	64.07	70.00	76.78	85.71	0.00	70.06
	outer	69.83	66.66	85.98	54.25	64.07	70.00	76.78	85.71	0.00	69.90
	inner	72.93	68.81	87.22	55.31	64.67	70.00	80.35	85.71	33.33	72.48
	exact	69.83	66.66	85.98	54.25	64.07	70.00	76.78	85.71	0.00	69.92
Teranishi+17 :+ext	whole	71.63	66.23	84.73	54.25	61.07	56.66	75.00	80.95	0.00	70.48
	outer	70.63	62.79	84.42	45.74	55.08	56.66	58.92	80.95	0.00	68.39
	inner	72.75	64.94	85.04	49.46	55.68	61.66	62.50	80.95	33.33	70.48
	exact	70.55	62.79	84.42	45.74	55.08	56.66	58.92	80.95	0.00	68.34
Teranishi+17	whole	66.59	63.87	78.50	52.65	53.89	50.00	78.57	85.71	33.33	65.98
Ficler+16	whole	65.08	71.82	74.76	17.02	56.28	51.66	91.07	80.95	33.33	64.14
Hara+09	whole	64.2	54.2	80.4	22.9	59.9	36.7	51.8	85.7	66.7	61.5

Teranishi+17, Ficler+16, Hara+09 の数値は論文で報告されている結果より抜粋.

6.3 考察

文単位での完全一致

並列構造単位での評価に加えて, 文単位での全並列構造の完全一致について評価した. 評価に際して, 文が持つ並列構造に応じて文を次の五つに分類する.

All: 並列構造を持つ全ての文.

- **Simple**: 二つの並列句から成るただ一つの並列構造を持つ文.
- **Complex**: Consecutive または Multiple に該当する文.
 - **Consecutive**: 三つ以上の並列句から成る並列構造を少なくとも一つ持つ文.
 - **Multiple**: 複数の並列構造を持つ文.

なお, Simple と Complex にそれぞれ該当する文の集合は互いに素であり, All はそれらの和集合であるが, Consecutive と Multiple の文集合には一部重複がある.

表 6 は PTB と GENIA における完全一致率の評価結果である. いずれのデータセットにおいても, Simple の文において提案手法はベースラインと比較して高い性能を得ている. このパフォーマンスの差は二つのシステムが学習している境界の違いに由来するものだと考えられる. Simple の文の場合, 提案手法は等位接続詞の前後に出現する二つの並列句の内側・外側境界を学習しているのに対し, ベースラインのシステムは並列構造の外側境界のみを学習・予測している. しかし, 同格や副詞句表現が等位接続詞と並列句の間に出現し得る場合があり, 二つの並列句が等位接続詞に隣接するという仮定は必ずしも成立しない. 提案手法は PTB において, Consecutive と Multiple の文についてもベースラインの精度を上回っている. ベースラインの手法は並列構造全体のスパンを予測してから個々の並列句に分割しているため, 準等位接続詞

表 6 文単位での並列構造の完全一致率

(単位:%)

文の種別		PTB (開発)	PTB (テスト)	GENIA
Ours	All	492/673 = 73.10	619/873 = 70.90	1555/2508 = 62.00
	· Simple	368/481 = 76.50	467/609 = 76.68	1088/1507 = 72.19
	· Complex	124/192 = 64.58	152/264 = 57.57	467/1001 = 46.65
	· Consecutive	45/ 66 = 68.18	60/ 96 = 62.50	170/ 358 = 47.48
	· Multiple	92/146 = 63.01	103/197 = 52.28	358/ 822 = 43.55
Teranishi+17 :+ext	All	468/673 = 69.53	577/873 = 66.09	1502/2508 = 59.88
	· Simple	358/481 = 74.42	444/609 = 72.90	1047/1507 = 69.47
	· Complex	110/192 = 57.29	133/264 = 50.37	455/1001 = 45.45
	· Consecutive	40/ 66 = 60.60	48/ 96 = 50.00	187/ 358 = 52.23
	· Multiple	78/146 = 53.42	92/197 = 46.70	328/ 822 = 39.90

表 7 異なる設定によるモデルの性能の違い (PTB, 開発)

(単位:%)

		全並列構造, exact			inner
		P	R	F	F
Ours	default	77.32	77.59	77.45	79.34
	-POS tags	72.47	72.64	72.55	75.61
	-CharCNNs	76.88	76.88	76.88	79.12
	-GloVe	75.64	75.82	75.73	77.62
	+ELMo	80.35	81.01	80.68	81.97
	+BERT	83.31	83.60	83.46	84.40
	concat feature	74.88	74.88	74.88	77.24
Teranishi+17 :+ext	default	74.13	73.34	73.74	75.63
	+ELMo	77.82	79.48	78.64	79.92
	+BERT	78.52	80.18	79.34	80.28

とならないようなカンマの出現により、誤って並列構造を分割し得る。対して提案手法はカンマなどの準並列キーが、実際に準等位接続詞として働いているかどうかについて予測しており、Consecutive の文をより高い精度で導出することができている。しかし GENIA では並列構造全体に占める名詞句の並列構造の割合が高く、かつ複合名詞が羅列するような単純な並列が多いため、カンマによる並列構造の分割が機能し、結果としてベースラインの手法が Consecutive の文において精度高く解析できていると推測される。

並列構造解析に有効な情報

並列構造解析に有効なベクトル表現・特徴計算について分析するため、異なる設定によって評価実験を行った。表 7 はその結果を示している。品詞タグがない場合にモデルの性能は顕著

に低下した. また品詞の他に, 文字 (形態) 情報が有効な素性であることも確認できた. これらが有効な情報である理由として, 特に名詞句の並列など類似度の高い並列句ペアにおいては, 品詞や接頭辞・接尾辞の一致が並列句の範囲同定の手がかりになるからだと推測している. また, 事前学習された単語の埋め込み表現がこのタスクにおいて有益であることも確認された. 近年広く利用されている文脈を考慮した単語分散表現を用いた場合, ELMo (Peters et al. 2018) では性能がわずかに上昇し, BERT (Devlin et al. 2019) では大幅に向上した¹³. ELMo は双方向型 LSTM により学習されており, LSTM の層ごとに前方向・後方向の文脈について独立に計算しているのに対し, BERT では Transformer (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin 2017) が用いられており, アテンションによって両方向の文脈を同時に見ているという違いがある. 我々のモデルではすでにエンコーダ内に双方向型 LSTM が用いられているが, アテンションのような単語対単語の関係を考慮するようなモデルは組み込まれていないため, Kurohashi and Nagao (1994) や Shimbo and Hara (2007) の手法で用いられているアラインメントと同様の働きを Transformer によって獲得できた可能性がある. 今後はエンコーダに Transformer を用いるなど, アーキテクチャによる性能差をより細かく調査したい. また外側境界スコア付与モデル内での特徴計算において, 内側境界スコア付与モデルと同様の特徴関数 (表 7 では *concat feature* として表記) を用いた場合, 性能が低下した. 式 (17) は並列句ペアの類似性・可換性をとらえる目的で設計されており, 単に b^l, e^r の二点だけを計算の考慮に入れた *concat feature* では, 外側境界のそのような特徴をとらえられていないと分析する.

ベースラインのモデルに ELMo や BERT によるベクトル表現を用いた場合, 提案手法の default の設定を上回る性能が得られた. ベースラインのモデルに ELMo・BERT を用いることで解けるようになった並列構造と, ベースラインの手法から提案手法に変更することで解けるようになった並列構造が仮に同一であった場合, 提案手法に ELMo・BERT を用いることによる改善は小さいはずである. しかしながら提案手法においても ELMo・BERT のベクトル表現を使用することによる性能改善は大きい. このことから, 手法の改善によって解けるようになった並列構造と, ELMo・BERT を使って解けるようになった並列構造は性質が異なると推測される. 各モデルによって解決された並列構造を表 8 に具体的に示す. 表 8 より, 提案手法で用いるアルゴリズムによって並列キーや準並列キーが複数出現する文において, 複雑な並列構造をより頑健に同定できるようになったのに対し, 文脈を考慮したベクトル表現は “NP and NP PP ...” や “VP and VP ADVP ...” のような系列に対して, PP や ADVP が後ろの並列句に含まれるかどうかといった前後文脈により依存するような曖昧な並列構造を同定するのに有効に働いてい

¹³ ELMo の設定では *Original* のモデル (https://s3-us-west-2.amazonaws.com/allennlp/models/elmo/2x4096.512.2048cnn.2xhighway/elmo.2x4096.512.2048cnn.2xhighway_weights.hdf5) を使用した.

BERT の設定では BERT-Large, uncased のモデル (https://storage.googleapis.com/bert_models/2018.10.18/uncased.L-24_H-1024_A-16.zip) を使用した. どちらのベクトル表現も次元数は 1024 次元である.

表 8 ベースライン+BERT と提案手法の解析結果の違い

(a) ベースライン+BERT の解析誤りが提案手法によって改善された例

1. Teranishi+17:+ext +BERT (誤り)	[Baseball] , [that game of the long haul] , [is the quintessential sport of the mean] , and [the mean ol' law caught up with the San Francisco Giants in the World Series last weekend] .
Ours (正解)	[Baseball , that game of the long haul , is the quintessential sport of the mean] , and [the mean ol' law caught up with the San Francisco Giants in the World Series last weekend] .
2. Teranishi+17:+ext +BERT (誤り)	“ [They 've been laggard] , [” he says] , [“] but [they 'll have to become more aggressive] . ”
Ours (正解)	“ [They 've been laggard] , ” he says , “ but [they 'll have to become more aggressive] . ”

(b) 提案手法の解析誤りがベースライン+BERT によって改善された例

3. Ours (誤り)	The rationale is that an interruption of trading will allow investors to reconsider their [strategies] , [calm sellers] and [lead buyers] to enter the market at indicated new price levels .
Teranishi+17:+ext +BERT (正解)	The rationale is that an interruption of trading will allow investors to [reconsider their strategies] , [calm sellers] and [lead buyers to enter the market at indicated new price levels] .
4. Ours (誤り)	They combined for [25 hits] , [six home runs] and [24 runs batted in in the five games against the Cubs] .
Teranishi+17:+ext +BERT (正解)	They combined for [25 hits] , [six home runs] and [24 runs batted in] in the five games against the Cubs .

ると考えられる。

解析結果の定性的分析

表 9 に提案手法による並列構造の解析結果の例を示す。例 1 は二つの独立した並列構造を正しく解析できており、特に後者の並列構造では the retail and financial sectors という名詞句に対し、the retail と financial sectors という二つの名詞句の並列ではなく、retail と financial の二つの形容詞の並列として解析に成功している。例 2 は自動詞と他動詞の二つの動詞句による不均衡な並列構造の範囲を正確にとらえられている。例 3 は三つの平叙節の並列を解析できており、またそのうち一つの並列句内に埋め込まれている並列構造の解析にも成功している。対して例 4 では、at prevailing market prices という前置詞句が or の直後の並列句に含まれるものとして解析している。解析結果の境界は句の切れ目として不自然ではないが、前置詞句の主辞 (buy) をとらえることができていない。例 5 では入れ子となっている並列構造は解析できてい

表 9 提案手法の並列構造解析器による出力例

(a) システムによって正しく解析された並列構造の例

1. 入力文	In addition to a general slowing of the computer industry , NCR , which sells automated teller machines and computerized cash registers , is also affected by the retail and financial sectors , “ areas of the economy that have generally not been robust , ” notes Sanjiv G. Hingorani , an analyst for Salomon Brothers Inc .
出力	[teller machines] and [computerized cash registers] ; [retail] and [financial]
2. 入力文	While profitable , it “ was n’t growing and was n’t providing a satisfactory return on invested capital , ” he says .
出力	[was n’t growing] and [was n’t providing a satisfactory return on invested capital]
3. 入力文	The Dow Jones Industrial Average jumped sharply yesterday to close at 2657.38 , panic did n’t sweep the world ’s markets , and investors large and small seemed to accept Friday ’s dizzying 190-point plunge as a sharp correction , not a calamity .
出力	[The Dow Jones Industrial Average jumped sharply yesterday to close at 2657.38] , [panic did n’t sweep the world ’s markets] , and [investors [large] and [small]] seemed to accept Friday ’s dizzying 190-point plunge as a sharp correction , not a calamity] .

(b) システムによって誤って解析された並列構造の例

4. 入力文	The company said it will buy additional shares “ from time to time in the open market or in private transactions at prevailing market prices . ”
出力	[in the open market] or [in private transactions at prevailing market prices]
正解	[in the open market] or [in private transactions]
5. 入力文	Dealers said morning activity was hectic as prices dropped in response to gains in the stock market and losses in Treasury securities , but trading slowed to moderate levels in the afternoon .
出力	[Dealers said morning activity was hectic as prices dropped in response to [gains in the stock market] and [losses in Treasury securities]] , but [trading slowed to moderate levels in the afternoon] .
正解	Dealers said [morning activity was hectic as prices dropped in response to [gains in the stock market] and [losses in Treasury securities]] , but [trading slowed to moderate levels in the afternoon] .

るが, that 節の that が省略されているために, Dealers said ...securities を一つの平叙節として認識してしまっている. このように, 提案手法によって出力された並列句は句の境界については不自然ではないものの, 並列構造の前後の句が並列句によって共有されるか否かの曖昧性を十分に解消できていない. 並列となる句が前後の句に対して共通の関係を持つかどうかを判別することが提案手法の今後の課題である.

他言語への適用可能性

本節では評価実験により提案手法の英語における有効性を示した。4節で示した構文解析を用いる本手法は、解析対象のテキストに対して以下の要件を前提としている。

1. テキストが文単位に分けられ、入力文は単語分割が完了したものとして単語から構成される。
2. (準) 等位接続詞となり得る語が定義されている。
3. (準) 等位接続詞が形成する並列構造のパターンが、文脈自由文法の規則として定義されている。

これらの前提条件が満たされる限り、CKY アルゴリズムによる構文解析の過程において、並列キーに対する並列構造の有無が判定され、並列句の開始・終了位置となるあらゆる単語の組み合わせが精査され、並列構造の組み合わせが木として導出される、という一連のプロセスが言語やドメインに依存することなく機能する。また5節で提案したニューラルネットワークのモデルは、単語に対する品詞タグや事前学習されたベクトル表現を必須としていないが、高精度な解析のためにはそれらのリソースが利用できることが望ましい。さらにニューラルネットワークの学習のためには、並列句の範囲が明示的に付与された文が大量に利用できることが想定される¹⁴。

以上の条件について、提案手法の日本語への適用を例に具体的に考える。『現代日本語書き言葉均衡コーパス』(小椋, 小磯, 富士池, 宮内, 小西, 原 2011) (BCCWJ) は、新聞、書籍、雑誌、白書、ブログ、掲示板など、様々なジャンルにまたがって収集されたテキストから成るコーパスであり、このコーパスでは一部のデータに対して並列構造の範囲が付与されている(浅原, 松本 2018)。並列構造の範囲は国語研短単位形態素境界に基づいて分割された文の構成要素を基準にして付与されており、六つのジャンルから成るコアデータでは総数 57,109 の文に 14,368 の並列構造が出現している。したがって前提条件 1 は解決され、学習に必要なアノテーションについても入手が可能であり、単語に対する品詞タグやベクトル表現もツールを使って付与することができる。また並列構造や品詞情報のアノテーションから、(準) 等位接続詞となり得る語を定義することができ、前提条件 2 も満たされる。前提条件 3 について、例えば、“本物ならではの [風合い], [風格] そして [高級感あふれる質感] は” という並列構造が表 1 の規則から導出できるように、提案規則を用いることで大部分の並列構造を導出することができると見込まれる。ただし、前終端記号に対する規則 (11) (12) については、条件 2 で設定した (準) 等位接続詞から定義される必要がある。日本語では “[A] と [B] と [C]” という並列構造のように、複数の等位接続詞から成る並列構造が頻出するが、例えば規則 (2) に類似した規則, “COORD →

¹⁴ 本実験で用いた GENIA Treebank beta では、並列構造を持つ 2,508 の文のなかに 3,598 の並列構造が出現していることから、学習のためには少なく見積もっても数千以上のオーダーでアノテーション済みの文が必要であろう。

CONJ CC COORD”を追加することで, そのような並列構造を導出できる. また日本語の法令文においては, 「並びに」と「及び」から成る並列構造が “[A] 並びに [[B] 及び [C]]” と一意に解釈できるように上位・下位の階層関係が定められており, このようなドメインや文書に固有のルールも文脈自由文法規則として取り入れることが可能である.

以上を踏まえて, 本手法を日本語に適用できる見込みは高く, 本手法の提案規則に基づいていくつかの拡張を伴うことで, 英語や日本語に類似した他言語においても本手法が適用可能であると期待される.

7 おわりに

本論文では, 並列構造解析のためのシンプルでより正確なフレームワークを提案した. 提案手法は並列構造解析のタスクを三つのサブタスクに分解し, それぞれに異なるニューラルネットワークを用いる. 並列構造を範囲の競合なく導出できる文脈自由文法規則を定義し, CKY アルゴリズムによる構文解析とニューラルネットワークを組み合わせることで, 我々のシステムは入力文に対する並列構造の集合を出力する. 英語における評価実験の結果, 並列構造の最適な組み合わせの導出のために三つのニューラルネットワークが協調して働いており, 既存のシステムやその拡張を上回る解析精度を達成していることが示された. また分析により, 提案手法はベースライン手法と比較して複雑な並列構造を文レベルで正確に解析できることが分かった. 我々のシステムは並列構造の範囲同定に誤る場合であっても, 並列句の境界は句の区切りと一致している傾向にあり, 範囲の候補として有力である. 今後の課題として, CKY アルゴリズムによる構文解析を拡張し, スコアが高いものから順に複数個の構文木を取り出し, 並列構造の前後の文脈を考慮した構文木候補のリランキング手法を開発することによって, 解析精度をより向上させることが挙げられる. また日本語を含む他言語に本手法を適用し, 評価実験を行うことも課題とする.

謝 辞

本研究は JST CREST (課題番号: JPMJCR1513) の助成を受けて行った. また本研究の一部は, The 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies で発表したものである (Teranishi, Shindo, and Matsumoto 2019).

参考文献

- Agarwal, R. and Bogges, L. (1992). “A Simple but Useful Approach to Conjunct Identification.” In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 15–21. Association for Computational Linguistics.
- Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). “Globally Normalized Transition-Based Neural Networks.” In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2442–2452. Association for Computational Linguistics.
- 浅原正幸, 松本裕治 (2018). 『現代日本語書き言葉均衡コーパス』に対する文節係り受け・並列構造アノテーション. 自然言語処理, **25** (4), pp. 331–356.
- de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). “Generating Typed Dependency Parses from Phrase Structure Parses.” In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pp. 449–454. European Language Resources Association (ELRA).
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics.
- Dozat, T. and Manning, C. D. (2017). “Deep Biaffine Attention for Neural Dependency Parsing.” In *Proceedings of the 5th International Conference on Learning Representations*.
- Eisner, J. M. (1996). “Three New Probabilistic Models for Dependency Parsing: An Exploration.” In *Proceedings of the 16th International Conference on Computational Linguistics (Volume 1)*, pp. 340–345.
- Ficler, J. and Goldberg, Y. (2016a). “Coordination Annotation Extension in the Penn Tree Bank.” In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 834–842. Association for Computational Linguistics.
- Ficler, J. and Goldberg, Y. (2016b). “A Neural Network for Coordination Boundary Prediction.” In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 23–32. Association for Computational Linguistics.
- Hanamoto, A., Matsuzaki, T., and Tsujii, J. (2012). “Coordination Structure Analysis using Dual Decomposition.” In *Proceedings of the 13th Conference of the European Chapter of*

- the Association for Computational Linguistics*, pp. 430–438. Association for Computational Linguistics.
- Hara, K., Shimbo, M., Okuma, H., and Matsumoto, Y. (2009). “Coordinate Structure Analysis with Global Structural Constraints and Alignment-Based Local Features.” In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 967–975. Association for Computational Linguistics.
- Hochreiter, S. and Schmidhuber, J. (1997). “Long Short-Term Memory.” *Neural Computation*, **9** (8), pp. 1735–1780.
- Hogan, D. (2007). “Coordinate Noun Phrase Disambiguation in a Generative Parsing Model.” In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 680–687. Association for Computational Linguistics.
- Kawahara, D. and Kurohashi, S. (2008). “Coordination Disambiguation without Any Similarities.” In *Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 425–432. Coling 2008 Organizing Committee.
- Kim, J. D., Ohta, T., Tateisi, Y., and Tsujii, J. (2003). “GENIA Corpus—A Semantically Annotated Corpus for Bio-textmining.” *Bioinformatics*, **19** (suppl 1), pp. i180–i182.
- Kingma, D. P. and Ba, J. (2015). “Adam: A Method for Stochastic Optimization.” In *Proceedings of the 3rd International Conference on Learning Representations*.
- Kurohashi, S. and Nagao, M. (1994). “A Syntactic Analysis Method of Long Japanese Sentences Based on the Detection of Conjunctive Structures.” *Computational Linguistics*, **20** (4), pp. 507–534.
- Ma, X. and Hovy, E. (2016). “End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF.” In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1064–1074. Association for Computational Linguistics.
- Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013). “Efficient Estimation of Word Representations in Vector Space.” In *Proceedings of the 1st International Conference on Learning Representations*.
- Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). “Universal Dependencies v1: A Multilingual Treebank Collection.” In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pp. 1659–1666. European Language Resources Association (ELRA).
- 小椋秀樹, 小磯花絵, 富士池優美, 宮内佐夜香, 小西光, 原裕 (2011). 『現代日本語書き言葉均

衡コーパス』形態論情報規程集 (第4版). 国立国語研究所.

- Pennington, J., Socher, R., and Manning, C. D. (2014). “Glove: Global Vectors for Word Representation.” In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543. Association for Computational Linguistics.
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). “Deep Contextualized Word Representations.” In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237. Association for Computational Linguistics.
- Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). “Learning Accurate, Compact, and Interpretable Tree Annotation.” In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 433–440. Association for Computational Linguistics.
- Resnik, P. (1999). “Semantic Similarity in a Taxonomy: An Information-based Measure and Its Application to Problems of Ambiguity in Natural Language.” *Journal of Artificial Intelligence Research*, **11** (1), pp. 95–130.
- Shimbo, M. and Hara, K. (2007). “A Discriminative Learning Model for Coordinate Conjunctions.” In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 610–619. Association for Computational Linguistics.
- Stern, M., Andreas, J., and Klein, D. (2017). “A Minimal Span-Based Neural Constituency Parser.” In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 818–827. Association for Computational Linguistics.
- Teng, Z. and Zhang, Y. (2018). “Two Local Models for Neural Constituent Parsing.” In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 119–132. Association for Computational Linguistics.
- Teranishi, H., Shindo, H., and Matsumoto, Y. (2017). “Coordination Boundary Identification with Similarity and Replaceability.” In *Proceedings of the 8th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 264–272. Asian Federation of Natural Language Processing.
- Teranishi, H., Shindo, H., and Matsumoto, Y. (2019). “Decomposed Local Models for Coordinate Structure Parsing.” In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*

- (*Long and Short Papers*), pp. 3394–3403. Association for Computational Linguistics.
- Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). “Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network.” In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pp. 173–180. Association for Computational Linguistics.
- Tsatsaronis, G., Schroeder, M., Paliouras, G., Almirantis, Y., Androutsopoulos, I., Gaussier, E., Gallinari, P., Artieres, T., Alvers, M. R., Zschunke, M., and Ngomo, A.-C. N. (2012). “BioASQ: A Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering.” In *AAAI Fall Symposium Series: Information Retrieval and Knowledge Discovery in Biomedical Text*, pp. 92–98.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). “Attention is All you Need.” In *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc.
- Yoshimoto, A., Hara, K., Shimbo, M., and Matsumoto, Y. (2015). “Coordination-Aware Dependency Parsing (Preliminary Report).” In *Proceedings of the 14th International Conference on Parsing Technologies*, pp. 66–70. Association for Computational Linguistics.

略歴

寺西 裕紀：2014 年慶應義塾大学商学部卒。同年，株式会社アイスリーデザイン入社。Web・モバイルアプリケーションの開発・運用に従事。2018 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了。同年，同大学院先端科学技術研究科情報科学領域博士後期課程入学。自然言語処理の研究に従事。

進藤 裕之：2009 年，早稲田大学先進理工学研究科博士前期課程修了。同年 NTT コミュニケーション科学基礎研究所入所。2013 年，奈良先端科学技術大学院大学情報科学研究科博士後期課程修了。2014 年より現在まで，奈良先端科学技術大学院大学情報科学研究科助教。これまで，主に自然言語処理の研究に従事。情報処理学会，人工知能学会，言語処理学会，ACL 各会員。

渡辺 太郎：1994 年京都大学工学部情報工学科卒業。1997 年京都大学大学院工学研究科情報工学専攻修士課程修了。2000 年 Language and Information Technologies, School of Computer Science, Carnegie Mellon University, Master of Science 取得。2004 年京都大学博士（情報学）。ATR および NTT, NICT にて研究員，また，グーグルでソフトウェアエンジニアとして勤めた後，2020 年より奈良先端科学技術大学院大学教授。自然言語処理や機械学習，機械翻

訳の研究に従事.

松本 裕治: 1977 年京都大学工学部情報工学科卒. 1979 年同大学大学院工学研究科修士課程情報工学専攻修了. 同年電子技術総合研究所入所. 1984~85 年英国インペリアルカレッジ客員研究員. 1985~87 年 (財) 新世代コンピュータ技術開発機構に出向. 1988 年京都大学助教授, 1993 年奈良先端科学技術大学院大学教授. 2020 年理研 AIP 知識獲得チームリーダー, 現在に至る. 工学博士. 専門は自然言語処理. 情報処理学会, 人工知能学会, 言語処理学会, AAAI, ACL, ACM 各会員. 情報処理学会フェロー. ACL Fellow.

(2020 年 5 月 1 日 受付)

(2020 年 7 月 1 日 再受付)

(2020 年 7 月 29 日 採録)