単語-文節間の変換による日本語係り受け解析

松野 智紀　松本 裕治
奈良先端科学技術大学院大学
{matsuno.tomoki.mr1, matsu}@is.naist.jp

1 はじめに

日本語の自然言語処理において，様々な基礎解析タスクは文節単位で行われてきた．しかし，近年 Universal Dependencies など複数の言語で一貫した解析を可能にしようとする流れのなかで，日本語においても既存の文節単位の係り受けコーパスを変換することにより単語単位の係り受け情報などを付与したデータセットが作成されている [1].

本稿では，係り受けを文節単位から単語単位に変換することで，文節のまとめ上げと文節間の係り受け解析を同時に行える可能性を示した．実験では，文節係り受けからルールベースで変換された単語係り受けのツリーバンクにおいて単語単位の係り受け解析のために開発された構文解析器を訓練し，性能評価を行った．

ツリーバンクの作成は，京大コーパス中の文に付与されている文節係り受け構造を変換することで行なった．孤に付与されるラベルは文節内の係り受けを示す inside，文節間の係り受けを示す outside と root の 3 種類とした．構文解析器は Dozat ら [3] の Deep Biaffine Parser を基礎とし，双アフィン分類器中の重み行列がそれぞれ正方行列，対称行列，巡回行列となるような制約を置くモデルで実験した [4]．ベースラインとして文節単位の係り受け解析器である CaboCha [9] および J.DepP [8] を用いた．評価には得られた単語単位の係り受け解析結果から元の文節単位の正解係り受けをどれだけ再現できるかを用いた．結果として巡回行列の制約に基づくモデルが単一のモデルとしての最高精度を達成した．また，文節内の係り受けを利用したチャンキングの評価を行い，今までパイプライン的に行われていた文節チャンキングと文節係り受けを同時に十分な精度で行える可能性を示した．

2 提案手法

2.1 文節単位から単語単位への変換

日本語の文節は内容語およびそれに続くオプショナルな機能語からなる．ここでは，文節中の単語列のうち，一番右の内容語を文節ヘッドと呼ぶ．文節中のその他の単語は全て文節ヘッドを親とし，各文節の文節ヘッドは自分以外の文節における文節ヘッドまたはルートを親とする．

図 1 は変換の一例である．ここで / は文節境界，下線は文節ヘッドを示す．まず，各文節について，文節ヘッドを定める．例えば 2 番目の文節において内容語は「えび，フライ」で機能語は「を」であるため，内容語の中で最も右にある「フライ」が文節ヘッドとなる．文節内係り受けについては，文節ヘッド以外の単語は所属する文節の文節ヘッドを親とする（図中では文節内係り受けを点線で表した）．各文節ヘッドは，所属する文節の親となる文節の文節ヘッドまたはルートを親とする（図中では文節間係り受けを実線で表した），

2.2 文節単位での評価

文節単位での解析精度は，文節境界を所与とした状態で，単語単位の解析結果から元の文節係り受けを再構築できるかを評価する．具体的には以下の 2 点を満たすとき，その文節から伸びる弧を正解とする．

1. 文節内の少なくとも 1 つの語が文節外の語に係る
2. 文節内のすべての単語が自身が所属する文節内か正解の係り先の文節内の単語に係る

図 2 に具体例を示す．2 番目の文節において，単語「えび」と「フライ」はそれぞれ別の語に係っているが，両方とも係り先は正解の文節中の語であるため，正解とみなされる．図 3 では,「フライ」は正解の文節中の語に係っているが,「えび」は別の文節中の語に係っているため，不正解とされる．

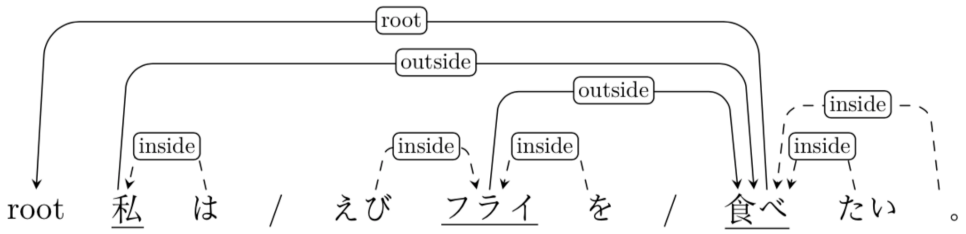

図 1: 文節単位係り受けから単語単位係り受けへの変換の例

3 関連研究

日本語の単語単位係り受けツリーバンク

これまでにいくつかの日本語単語単位係り受けのツリーバンクが公開されている． それらのほとんどは文に単語単位の係り受け構造を付与するかたちではなく，既存の文節単位の係り受け構造が付与されたデータセットからルールベースで単語単位の係り受けに変換することによって作られている．

UD_Japanese-KTC[1] は，田中らによって京大コーパスの一部を変換および修正して作られた句構造ツリーバンク [7] を係り受け構造に変換することで作られている．京大コーパスに基づく単語単位係り受けツリーバンクという点では我々の実験設定と同じだが，我々の実験設定は京大コーパスの全部を利用している点と文節境界を陽に扱っている点でこのツリーバンクとは異なっている．

UD_Japanese-GSD[1]は，日本語 Wikipedia の文をトークナイザーで分割後，文節単位の構文解析器によって解析し，Universal Dependencies のルールに沿って単語単位に変換することで作られた．

UD_Japanese-PUD[2]は，複数の言語にまたがるパラレルコーパスであり，UD_Japanese-GSD と同じ手法で作られた．

UD_Japanese-BCCWJ[6] は，現代日本語書き言葉均衡コーパスをスクリプトによって自動で変換して作られた．Universal Dependencies のツリーバンクの中では 2 番目に大きいツリーバンクである．

UD_Japanese-Modern[5] は，近代日本語コーパス（CHJ）に基づいたツリーバンクで，UD_Japanese-BCCWJ の係り受けスキームを使って文節係り受けをアノテーションした後，同ツリーバンクに使用された変換スクリプトを用いて単語単位係り受けに変換された．

4 実験

提案手法の評価および既存の日本語係り受け解析器との比較を行う．単語単位係り受けの解析には Dozat らによる Deep Biaffine Parser[3] を用いた．モデルは Dozat ら の実装を基本とした[3]．

4.1 実装詳細

データ 実験には京大コーパス 4.0 を用いた．Yoshinaga ら [8] に従い, 次に示す分割でモデルの訓練・評価を行った．

- **訓練**: 1 月 1-11 日の記事と, 1 月から 8 月の社説.
- **開発**: 1 月 12-13 日の記事と, 9 月の社説.
- **評価**: 1 月 14-17 日の記事と, 10 月から 12 月の社説.

実験で用いる単語境界ならびに文節境界には京大コーパスの正解アノテーションを利用する. 単語の埋め込みベクトルには下記の Web サイトで公開されている訓練済み単語ベクトル[4]を使用した．LSTM の隠れ層の次元を 400，arc 分類器の次元を 500 に変更した以外は Dozat らの設定に従った. これらの変更は Dozat らが English Penn Treebank で行なった実験 [2] に基づいている．

4.2 重み行列の対称性および巡回性の制約に基づく手法

松野ら [4] は構文解析タスクにおいて，双アフィン分類器の双線型項に重み行列が対称行列または巡回行列であるような制約を加え，モデルのパラメータ数を削減しつつ解析精度が向上することを示した．そのため，本論でもこれらの手法を用いたモデルを実験する．

[1]http://universaldependencies.org/treebanks/ja_gsd/index.html
[2]http://universaldependencies.org/treebanks/ja_pud/index.html
[3]https://github.com/tdozat/Parser-v2
[4]https://github.com/Kyubyong/wordvectors

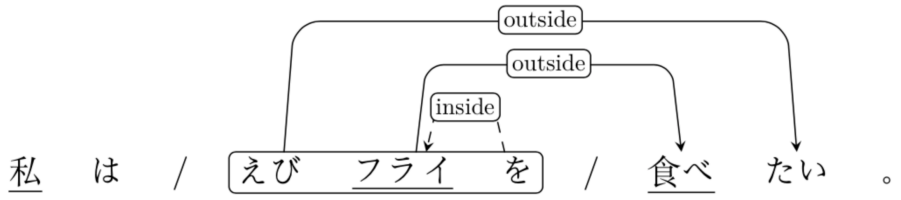

図 2: 係り受けの評価の例（正解）

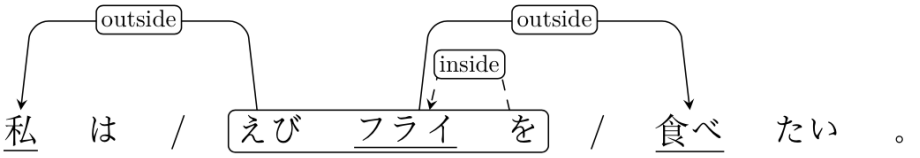

図 3: 係り受けの評価の例（不正解）

4.3 比較手法

比較手法として，代表的な日本語構文解析器の CaboCha [9] と J.DepP [8] を選んだ[5]．CaboCha は上述のデータ分割に基づいて訓練，評価を行なった．J.DepP は作者の Web ページに公開されている数値を引用した[6]．これらは遷移型の構文解析器であり，文節から抽出した組み合わせ素性によって各動作を選択する．

4.4 結果

文節係り受け

表 1 に結果を示す．評価は文節単位で行なった．提案手法はどれも 2 つの比較手法を上回っており，そのなかでも巡回行列に基づく手法が最高精度を記録している．

	精度
正方行列	92.57
対称行列	92.57
巡回行列	**92.71**
J.DepP	92.29
CaboCha	91.84
*J.DepP + KNP4.16	92.92

表 1: 文節係り受けの精度．比較手法の評価基準に習い, 最も右の文節については係り受けの評価から除外した．二重線以下は参考記録（脚注 5 参照）．

文節チャンキング

文節内係り受けを利用して文節チャンキングの性能を評価した．提案手法には正方行列に基づくモデルを用いた．比較手法には CaboCha を用いた．結果を表 2 に示す．チャンキングの性能は比較手法を大きく上回った．

これらの結果は，文節チャンキングのために分離した処理をせずとも，文節のまとめ上げと文節係り受け解析を同時に行うことにより，十分な性能が得られる

[5]J.DepP の作者の Web サイトでは J.DepP に KNP4.16 および KNP2.0 を加えたモデルの性能が公開されているが，前者は大規模なテキストから構築された格フレームを利用しており，後者はハイパーパラメータ設定が記載されていなかった．これらとの比較は公平ではないため，今回はモデル単体の性能を比較した

[6]http://www.tkl.iis.u-tokyo.ac.jp/ ynaga/jdepp/

	CaboCha	提案手法
F 値	95.70	99.02

表 2: 文節チャンキングの評価.

	使用する素性	精度
正方行列	全素性	92.57
	単語表記のみ	91.98
CaboCha	全素性	91.84

表 3: 提案手法を単語表記のみで訓練する実験.

ことを示唆している.

5 分析

5.1 LSTM による素性抽出

LSTM による素性抽出は人手による細かな素性設計に大きく依存しないことが知られている．そこで正方行列に基づくモデルにおいて素性として単語表記のみを用いる実験を行なった（表 3）．結果として，91.98%の精度を得た．単語表記以外の素性も用いる設定と比べると精度が 0.59 ポイント程度下回ったが，興味深いことに提案手法の 1 つである CaboCha と比べると精度が 0.14 ポイント上回っている．これは LSTM によって単語表記列のみからであっても係り受け解析に必要な素性が効率的に抽出できていることを示している.

5.2 文節間係り受けと文節内係り受け

我々が単語単位の係り受けを付与するときに，孤に付与するラベルとして，文節間係り受けにあたる孤には outside，文節内係り受けにあたる孤には inside を使った．表 4 にラベルごとの LAS を示す.

inside は高いスコアを達成している．これは，文節単位係り受けから単語単位係り受けへの変換ルールを構文解析器が学習できていることを示唆している.

文節間係り受けも，ベースラインの 2 つのモデルの文節係り受け解析精度と比べて妥当な性能を示している．これは，文節単位係り受け問題を単語単位係り受け問題の一部として解く手法が徒らに問題を複雑にしていないことを示している.

6 おわりに

本稿では, 文節係り受け構造の付与されたツリーバンクを単語係り受けに変換して解くことで，文節チャンキングと文節係り受け解析を十分な性能で同時に行える可能性を示した．文節係り受けの評価では，それらで訓練された構文解析器が元の文節係り受けをどれだけ解けたかを評価した．結果として，京大コーパスにおける実験で文節係り受け解析タスクにおける単一のモデルとしての最高精度を達成した．また，単語の表記のみを用いて解析器を訓練する実験では，細かな素性を用いる CaboCha と比べて高い精度を記録した.

	inside	outside	root
LAS	99.44	92.41	99.98

表 4: 文節内係り受けと文節間係り受けにおけるパフォーマンス.

参考文献

[1] Masayuki Asahara, Hiroshi Kanayama, Takaaki Tanaka, Yusuke Miyao, Sumire Uematsu, Shinsuke Mori, Yuji Matsumoto, Mai Omura, and Yugo Murawaki. Universal Dependencies Version 2 for Japanese. In *Proc. of LREC 2018*, Miyazaki, Japan, May 7-12, 2018 2018. European Language Resources Association (ELRA).

[2] Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734, 2016.

[3] Timothy Dozat, Peng Qi, and Christopher D. Manning. Stanford's graph-based neural dependency parser at the conll 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[4] Tomoki Matsuno, Katsuhiko Hayashi, Takahiro Ishihara, Hitoshi Manabe, and Yuji Matsumoto. Reduction of parameter redundancy in biaffine classifiers with symmetric and circulant weight matrices. *CoRR*, abs/1810.08307, 2018.

[5] M Omura, Y Takahashi, and M Asahara. Universal Dependency for Japanese Modern. In *JADH-2017*, 2017.

[6] Mai Omura and Masayuki Asahara. Ud-japanese bccwj : Universal dependencies annotation for the balanced corpus of contemporary written japanese mai omura and masayuki asahara national institute for japanese language and linguistics. 2018.

[7] Takaaki Tanaka and Masaaki NAGATA. Constructing a practical constituent parser from a japanese treebank with function labels. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 108–118. ACL, 2013.

[8] Naoki Yoshinaga and Masaru Kitsuregawa. Polynomial to linear: Efficient classification with conjunctive features. In *Pro. of the 2009 Conference on EMNLP: Volume 3 - Volume 3*, EMNLP '09, pages 1542–1551, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[9] 工藤 拓 and 松本 裕治. チャンキングの段階適用による日本語係り受け解析. 43(6):1834–1842, 2002.