短単位品詞の用法曖昧性解決と依存関係ラベリングの同時学習

松田 寛 * 大村 舞 † 浅原 正幸 †
株式会社リクルート Megagon Labs, Tokyo, Japan *
国立国語研究所 †
hiroshi_matsuda@megagon.ai {mai-om, masayu-a}@ninjal.ac.jp

1 はじめに

近年、オープンソース・ソフトウェア（以下 OSS）として **Stanford Core-NLP**[1]や **spaCy**[2]のような高機能な NLP フレームワークが利用可能となっている。これらは商用利用も可能[3]なライセンス形態で供与されている。特に商用アプリケーションでは i18n 対応コストが重視されることが多く、NLP フレームワークには（プログラムを書き換えることなく）リソース切り替えのみで様々な言語に対応可能であることが要請される。Stanford Core-NLP や spaCy では英語以外の多くの言語リソースが提供されているが、日本語には未対応の状況が長く続いており、日本国内での NLP フレームワーク普及促進を妨げる要因となるばかりでなく、データサイエンス領域における日本語のプレゼンス低下に繋がることが懸念される。

本稿では **Universal Dependencies** (Zeman[1]) (UD)に基づいて設計された spaCy を NLP フレームワークとして採用し、その日本語版リソースの実現に不可欠な学習系・解析系の機能実装と精度評価を行う。UD に基づく正解コーパスには現代日本語書き言葉均衡コーパス **BCCWJ** (Maekawa[2])を UD 化した **UD-Japanese BCCWJ** (Omura[3])を用いる。

日本語の平文を UD に基づいてトークン化するには形態素解析器が必要となる。spaCy は Python ライブラリとして提供されるため、本稿では形態素解析器 **Sudachi** (Takaoka[4])の Python クローンである **SudachiPy**[4] を使用することで言語リソースの Pure Python 化を実現する。Sudachi の辞書は **UniDic** 短単位品詞体系 (伝[5])をベースとするため、UniDic 体系に基づいて設計された UD-Japanese BCCWJ との親和性は高い。ただし、UD-Japanese BCCWJ の構築には後述のように UniDic 長単位品詞の参照が必要となるため、UniDic 短単位品詞体系に含まれる可能性に基づく品詞の解決（短単位品詞の用法曖昧性解決）が必要となる。本稿では依存関係ラベルに正解品詞を埋め込むことで、短単位品詞の用法曖昧性解決と依存構造解析を同時学習する方式を提案・評価する。

2 依存関係ラベルへの品詞埋め込み

2.1 UniDic 長単位品詞の推定

UniDic 短単位品詞体系では可能性に基づく品詞[5]が採用されている。Omura[3]では BCCWJ で短単位(SUW)・長単位(LUW)・文節の各単位長に対して付与される品詞等の情報を組み合わせて UD-Japanese BCCWJ を構築する手順が示されている。その中で、可能性品詞が付与されたトークンについて、人手で文脈を考慮して付与された LUW 品詞を用いて UD 品詞を決定している点は注意を要する。解析系の実装では正解の LUW 品詞を参照することができないため、可

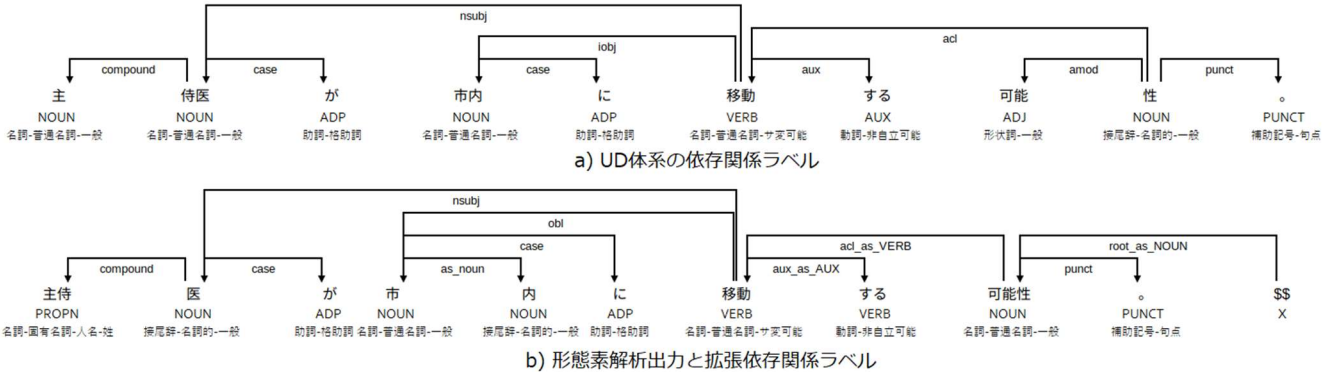

図 1 依存関係ラベルへの正解品詞の埋め込み

[1] https://stanfordnlp.github.io/CoreNLP/
[2] https://spacy.io/
[3] spaCy は MIT ライセンス、Stanford Core-NLP の商用ライセンスは個別契約が必要（2018 年 12 月時点）
[4] https://github.com/WorksApplications/SudachiPy
[5] 工藤拓 著「形態素解析の理論と実装」第 2 章 言語資源 に詳しい

能性品詞を含む形態素解析出力から UD 品詞を推定する必要がある。例えば、「名詞-普通名詞-サ変可能」の可能性品詞をもつトークンの UD 品詞は、文脈を考慮した上で NOUN または VERB のいずれかの UD 品詞に解決する必要がある。

2.2 語の長さの基準の統合

一般に、正解コーパスに対して形態素解析器の出力は品詞だけでなくトークン区切りにも異なりが生じる。その原因には単純な解析誤りも含まれるが、形態素解析辞書の語の長さの認定基準が正解コーパスのそれと異なる場合がある。その場合、何らかの変換ロジックを用いて整合性を確保する必要がある。例えば、つつじ (松吉[6])の階層的機能表現辞書のような形で変換ルールを実装して用いることもできる。

本稿では、単一の正解トークンの両端と複数の出力トークンの両端が対応する場合（断片化）や、単一の出力トークンの両端と複数の正解トークンの両端が対応する場合（単一化）は、正解もしくは出力いずれか長い側のトークンに統一して訓練を行う方針とする。対して、正解トークンと出力トークンの境界が一致しない区間については、統合が難しいため正解トークンをそのまま使用して訓練を行う。

2.3 品詞の補正

依存構造解析器の訓練において、コーパスに付与された正解品詞をそのまま使用するよりも、形態素解析器の出力品詞に置き換える方が依存関係ラベリングの単体精度を向上できることがある。本稿では正解トークンの品詞を形態素解析器の出力品詞に置き換えるだけでなく、出力品詞の補正まで含めた学習モデルも実装し、品詞まで含めたシステム全体での精度最大化を目指す。

3 提案方式

spaCy は正解コーパスからモデルを学習可能な Dependency Parser を備えており、その実装には CNN 学習器が使用されている[6, 7]。spaCy の依存関係ラベルは UD 準拠であるが、アプリケーション側でも任意に拡張可能となっている[8]。提案方式はこの拡張性を利用する。

3.1 短単位品詞の用法曖昧性解決

提案方式では、図 1 のように訓練コーパスに付与される UD 品詞を依存関係ラベルに埋め込むことで、文脈を考慮した可能性品詞の解決と依存構造解析を同時に行う。処理手順は、解析時に UD 品詞 `POS(Ti)` を復元するために、正解トークン `Ti` の依存関係ラベルを次のルールで書き換える[9]。

```
Label′(Ti)= ″{Label(Ti)}_as_{POS(Ti)}″
  POS′(Ti) = POS(Oj)
```

これと同時に、訓練コーパスを SudachiPy で再解析し、正解トークン `Ti` の品詞を `Ti` に対応する出力トークン `Oj` の UD 品詞 `POS(Oj)` で置き換える[10]。

解析時には `″{label}_as_{part_or_speech}″` の形式に拡張された依存関係ラベルを持つトークンの品詞を、ラベルに埋め込まれた UD 品詞に置き換えることで可能性品詞の解決を可能にする。

また、依存関係ラベルに UD 品詞を埋め込む方式を可能性品詞以外の品詞を持つトークンに適用することにより、形態素解析の品詞誤りを依存構造解析後の品詞置換処理で補正できる可能性がある。しかし、訓練時に全てのトークンの依存関係ラベルに正解品詞を埋め込む予備実験の結果では、逆に品詞判定精度が低下した。これは、依存関係ラベルのバリエーション増加により問題が複雑化したためと考えられる。

提案方式では短単位品詞の用法曖昧性解決に加えて、形態素解析出力品詞が正解と異なるトークンについても依存関係ラベルへの UD 品詞の埋め込みを行う。

3.2 双方向のトークン纏め上げ

Sudachi は UniDic 短単位品詞体系に基づく形態素辞書を持つが、利用用途に応じて複合語の単語長を A,B,C の三段階に切り替えて解析できる特長を備える (坂本[7])。モード A は UniDic 短単位相当、C は固有表現相当、B は A と C の中間的な単位である。この解析モードが影響しない文脈において、Sudachi と UD-Japanese BCCWJ の間で単語認定基準に一定の差異が存在する。特に数詞に関しては、BCCWJ が文字単位でトークン化しているのに対して、Sudachi は数字

[6] https://explosion.ai/blog/deep-learning-formula-nlp

[7] https://github.com/explosion/spaCy/issues/1057

[8] Token の tag_や pos_の追加には制限がある。UniDic の詳細な品詞・活用の保存にはトークン拡張フィールドを用いている。

[9] トークン区切りが完全に一致する場合のみ

[10] 形態素解析出力は品詞変換表を用いて仮の UD 品詞を割り当てておき、依存構造解析後に拡張ラベルの UD 品詞で置き換える。

が連続する区間をまとめた一つのトークンを出力する。このように形態素解析器と訓練コーパスの間には無視できない割合で基準が異なるトークンが存在するため、学習系と解析系の間でトークン長の調整が必要となる。

提案方式では、訓練コーパスを SudachiPy（モード C）で再解析し、正解トークンが SudachiPy の出力で断片化される場合は正解トークンを出力トークン列に置換する。その際、出力トークン列内部の依存関係には纏め上げ専用ラベルを"as_{part_or_speech}"の形式で設定する。これは、前節で述べた UD 品詞を依存関係ラベルに埋め込む方法のシームレスな拡張と言える。解析時は"as_{part_or_speech}"の形式の依存関係ラベルが連続する区間について纏め上げ処理を適用することで、トークン長の調整を実現する。逆に、正解トークン列が SudachiPy の出力で単一化される場合は、正解トークン列を出力トークンで置き換える。その際、置き換え後のトークンの品詞は元の正解トークン列のうち主辞に相当するトークン[11]の UD 品詞とする。

3.3 ルートラベルの拡張

提案方式では可能性品詞の解決のために依存関係ラベルを拡張して学習を行っている。しかし、spaCy では依存木の根となるトークン（ルート）に付与される依存関係ラベルは"root"に固定されているため、提案方式はそのままでは適用できない。そこで、文末に仮想トークンを追加し、真のルートから仮想トークンに"root_as_{part_or_speech}"の形式のラベルを用いて依存関係を追加することで、ルートについても依存関係ラベルの拡張を行う。

4 実験

UDPipe(Straka[8])をベースラインとして提案方式の精度と比較する。形態素解析器には株式会社ワークスアプリケーションズの SudachiPy および Sudachi 辞書 version 1.2.0 を、依存構造解析器には ExplosionAI UG の spaCy version 2.0.17 の CNN Dependency Parser を、Word Embedding には BCCWJ 全文と森羅プロジェクト(関根[9])の日本語 Wikipedia 本文全体を Sudachi で分かち書きし gensim version 3.6.0 の Skip-gram 実装を用いて次元数 100 で学習したものを用いる。CNN Dependency Parser は mini-batch size=128 の設定で学習を行う。またエポック毎に開発コーパスで精度スコアを評価し、それまでの精度スコアの最良値を連続して 3 回超えない場合に学習を打ち切る。実験で使用した UD-Japanese BCCWJ version 2.3 の構成を表 1 に示す。

表 1 UD-Japanese BCCWJ コーパスの構成

	訓練用[12]	開発用	評価用
文	40,686	8,427	7,881
トークン	916,915	180,767	168,759
可能性品詞に該当	167,220	31,912	30,095
Sudachi Mode C の区切りとの比較			
一致	841,312	167,222	155,133
仮 UD 品詞まで一致	690,835	137,901	128,108
単一化	80,902	13,324	13,323
断片化	1,061	146	243
不一致	486	75	60

表 2 正解トークンを使用した依存構造解析精度

	LAS	UAS	ROOT
Baseline (origin)	0.877 (0.320)	0.899 (0.431)	**0.968**
Baseline (115 sentences removed)	0.880 (*0.339*)	0.902 (0.458)	0.965
spaCy (115 sentences removed)	**0.883** (0.332)	**0.913** (*0.485*)	0.960

表 3 トークン化を含めた依存構造解析精度

LAS	UAS	LPOS	UPOS	POS	GAP
Token Recall					
0.879	0.910	0.873	0.899	0.982	0.998
Token Precision					
0.879	0.909	0.873	0.898	0.982	0.997
Sentence (ROOT=0.961)					
0.331	0.482	0.300	0.413	0.741	0.978

4.1 依存構造解析単体での精度評価

最初に UD-Japanese BCCWJ コーパスの正解トークンをそのまま使用する（依存関係ラベルを拡張しない）場合の精度を表 2 に示す（括弧内は文レベル精度）。この実験ではベースラインが用いている UDPipe と提案方式 (spaCy) の Dependency Parser の基本精度を比較するためのものである。なお、ベースラインでは依存関係の交差を含む文も学習に使用しているが、spaCy は非交差制約を前提とするため、訓練コーパスから交差を含む 115 文・6846 トークンを除外して学

[11] 外部から正解トークン列へのアークが 1 つの場合は外部アークの依存先を主辞とする。外部からのアークが複数の場合、それらの依存元が同一トークンの場合のみ正解トークン列末尾を主辞として単一化する。

[12] 訓練コーパスのみ、交差を含む 115 文を除外している

習した結果も併せて示す。

結果は UAS (Unlabeled Attachment Score)・LAS (Labeled Attachment Score) ともに提案方式が UDPipe を上回るが、ルート判定精度は逆にやや低い。

4.2 トークン化を含めた精度評価

提案方式で形態素解析器によるトークン化を行った場合の精度を表 3 に示す。LPOS・UPOS は品詞が正解である場合(POS)の LAS・UAS である。

提案方式は正解トークンを用いた場合に迫る良好な精度を示している。ただし、提案方式の評価では、正解トークンの区切りを出力トークンに合わせて長単位化しているため、区切り精度が高まる方向へのバイアスを考慮する必要がある。

4.3 可能性品詞の曖昧性解決精度

本節では可能性に基づく品詞の出現文脈に限定して品詞推定精度を評価する。ただし、評価対象は可能性品詞をもつ 30,095 トークンのうち、正解トークンと出力トークンの両端が一致するものに限定する。結果を表 4 に示す。また参考として、訓練コーパスの正解トークンを直接使用して学習したモデルを SudachiPy と組み合わせて解析した直接学習の結果も併せて示す。

表 4 可能性品詞の UD 品詞推定精度

	評価対象トークン数	UD 品詞推定精度
提案方式	28,104	0.965
直接学習	28,116	0.287

結果から提案方式は可能性品詞を高精度に UD 品詞に変換できている。直接学習の精度は非常に低いが、これは SUW 品詞から UD 品詞への変換に用いた品詞変換表の適合性が低いことが原因である。この品詞変換表は提案方式の素性生成を目的に作成したものを流用しており、SUW 品詞別に最頻の UD 品詞を求めるなど単純なチューニングを施すことで精度が大きく向上する余地がある。

一方、このように品詞変換表の適合性が低い状況においても、提案手法は拡張された依存関係ラベルから高い精度で UD 品詞を推定可能といえる。

5 結論

多言語 NLP フレームワークを UD ベースで日本語化する一つの方式として、学習コーパスと形態素解析器で語の長さの基準と品詞体系の両方が異なる組み合わせでも学習・解析を可能とする拡張された依存関係ラベルを用いる方式を提案・実装した。提案方式の解析精度は、形態素解析器でトークン化した場合でも正解トークンを利用する場合に近い水準であることを確認した。なお、本稿で実装した spaCy 日本語版リソースは OSS として公開する予定である。

謝辞

spaCy は精度のみならず動作安定性・ドキュメント品質にも優れたプロダクトです。spaCy を OSS として提供する ExplosionAI UG に敬意を表します。また SudachiPy の組み込みでは株式会社ワークスアプリケーションズの高岡一馬氏にご協力いただきました。ここにお礼申し上げます。最後に、多くの助言をいただきました Universal Dependencies コミュニティおよび国立国語研究所の皆様にお礼申し上げます。

参考文献

[1] Daniel Zeman, et al. Multilingual Parsing from Raw Text to Universal Dependencies, *Proceedings of the CoNLL 2017 Shared Task*.

[2] Kikuo Maekawa, et at. Balanced corpus of contemporary written Japanese. *Language Resources and Evaluation*, 48:2, pp 345–371. 2014.

[3] Mai Omura and Masayuki Asahara. UD-Japanese BCCWJ: Universal Dependencies Annotation for the Balanced Corpus of Contemporary Written Japanese. *Proceedings of the Second Workshop on Universal Dependencies*, pp 117-125. 2018.

[4] Kazuma Takaoka, et al. Sudachi: a Japanese Tokenizer for Business. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, 2018.

[5] 伝康晴,小木曽智信,小椋秀樹,山田篤,峯松信明,内元清貴,小磯花絵. コーパス日本語学のための言語資源: 形態素解析用電子化辞書の開発とその応用. 国書刊行会, pp 101–123. 2007.

[6]松吉俊,佐藤理史,宇津呂武仁. 日本語機能表現辞書の編纂. 自然言語処理, Vol.14, No.5, pp 123-146. 2007.

[7] 坂本美保,川原典子,久本空海,髙岡一馬,内田佳孝. 形態素解析器『Sudachi』のための大規模辞書開発. 言語資源活用ワークショップ 2018 発表論文集, pp 118-129. 2018.

[8] Milan Straka and Jana Strakova. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. *Proceedings of the CoNLL 2017 Shared Task*, pp 88–99.

[9] 関根聡,小林暁雄,安藤まや. Wikipedia 構造化プロジェクト「森羅 2018」, 言語処理学会 第 29 回年次大会 発表論文集, 2019.