テキスト生成モデルによる日本語形態素解析

児玉 貴志 [1]　植田 暢大 [1]　大村 和正 [1]
清丸 寛一 [1]　村脇 有吾 [1]　河原 大輔 [2]　黒橋 禎夫 [1]
[1] 京都大学　[2] 早稲田大学
{kodama,ueda,omura,kiyomaru,murawaki,kuro}@nlp.ist.i.kyoto-u.ac.jp
dkw@waseda.jp

概要

統合的日本語解析器 KWJA 1.0 は形態素解析を汎用言語モデルに基づくトークン分類問題として定式化した．しかし単語の読み，原形等はラベル集合が開放的であり，分類問題としては扱いづらい．本研究ではそうしたタスクをテキスト生成モデルで解く手法を提案する．提案手法は実装とタスク設計を単純化しつつ，特に読みと代表表記の推定で既存の形態素解析器を上回る精度を示した．提案手法は KWJA[1)] の形態素解析モジュールとして実装する予定である．

1　はじめに

BERT [1] 等の汎用言語モデルは言語の基礎的な解析においても威力を発揮している [2, 3]．汎用言語モデルの強力な文脈処理能力を利用すると，単語分割や品詞付与等の日本語形態素解析のサブタスクは単純なトークン分類問題として定式化できる．単語分割であれば各文字に B，I を，品詞付与であれば各サブワードに品詞ラベルを付与すれば良い．この手法は辞書引きに基づいてラティスを構築する従来手法と比べて単純だが，高い精度を達成している．

しかし，形態素解析のすべてのサブタスクがトークン分類問題として定式化しやすいわけではない．読み，原形等はラベル集合が開放的だからである．図 1 に示すように，入力サブワード（「京都」）に対応する読みのサブワード（「きょうと」）が存在しなかった場合，複数のサブワード（「きょう」，「と」）で読みを表現する必要があり，入力と出力が一対一対応の分類モデルでは対処できない．単語の原形や代表表記[2)] でもこうした問題は存在する．統合的日本語解析器である KWJA 1.0 [3][3)] ではこうした問題に対して，タスクごとにルールや辞書引き等で個別に対応しており，開発の負担や新語への対応等に課題が残る．

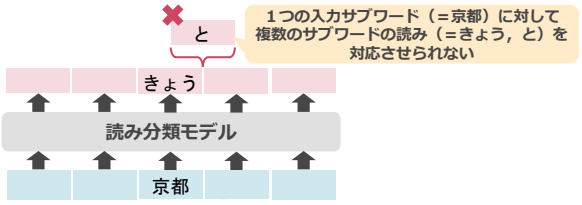

図 1　サブワードによる読み推定の誤り例

本研究では，トークン分類問題としては扱いづらい形態素解析の一部のサブタスクを，注釈付与コーパスから直接的に学習したテキスト生成モデルで解く手法を提案する．具体的には，入力文から見出し語（分かち書きされた単語），読み，原形，代表表記を所定のフォーマットに基づき end-to-end で生成する．エンコーダ・デコーダモデルを用いたテキスト生成は，サブワードの入出力が一対多（もしくは多対一）となっている場合でも対処できる．また，タスクごとに固有の処理を行う必要がなく，実装を単純化できる．

実験の結果，提案手法は実装とタスク設計の単純化に成功しつつ，分類モデルを用いた従来の解析器より，特に読みと代表表記を高い精度で推定できることが分かった．ただ，入力文に含まれない文字列が見出し語として解析結果に出てきてしまう例が少数だが確認された．そこで，見出し語を生成する際は入力文に含まれる文字列（サブワード）のみを生成するようにモデルの出力を補正する，見出し語強制デコーディングを提案する．この手法により提案手法のさらなる改善が見られた．

さらに，かな表記入力の曖昧性解消についても検

1)　https://github.com/ku-nlp/kwja
2)　各単語に対して与えられた ID．同一の語の表記揺れには同一の代表表記が与えられる．代表的な表記とその読みのペアで「京都/きょうと」のように表記される．
3)　執筆時点でのバージョンは 1.2.2 であり，このバージョンの KWJA を KWJA 1.0 と表記する．

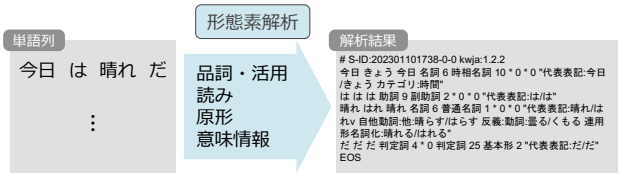

図 2　KWJA の形態素解析モジュールの概要

討する．例えば「こうえん」は文脈によって「公園」と「講演」のどちらにもなりうる．コーパス中にはこういったかな表記の単語が少ないため，漢字表記の単語を曖昧性のあるかな表記に置き換えることで疑似データを作成し，学習・評価に使用した．実験の結果，置き換えた単語について曖昧性解消の精度が大きく向上することを確認した．

2　関連研究

日本語形態素解析は辞書引きに基づいてラティスを構築する手法 [4, 5] が主流であったが，汎用言語モデルの登場にともない，単語（サブワード）列の各トークンにラベルを付与するトークン分類問題として解く手法が実用的な精度を達成している [3]．

汎用言語モデルに基づく統合的日本語解析器 KWJA [3] は入力誤り訂正から談話関係解析までを統合的に扱うが，このうち形態素解析モジュールの概要を図 2 に示す．単語列を入力とし，その単語に対する品詞・活用（品詞，品詞細分類，活用型，活用形の 4 つ），読み，原形，意味情報[4]を付与する．品詞・活用の付与はトークン分類問題を解くだけで完結する．ただし，読みは単語ではなくサブワードごとに推定している．この訓練データを作成するために，コーパスに単語単位で付与されている読みをサブワード単位に分割するアラインメント処理を動的計画法を用いて事前に行っている．原形は活用型，活用形をもとに活用変化表を参照し，正規化見出し語[5]からルールで出力している．また，意味情報は読み，原形，品詞，品詞細分類，活用型をキーとして，あらかじめ作成しておいた辞書を引いて，マッチした辞書項目の意味情報を出力している．本研究では，トークン分類問題として定式化すると煩雑な処理が必要となるこうしたサブタスクを，テキスト生成モデルを使用した単純な手法で解く．

トークン分類問題をテキスト生成モデルで解く研究はこれまでにも存在する．T5 [6] は要約や翻訳のほかに，分類問題も text-to-text で解けるように設計されたテキスト生成モデルである．また，GPT-3 [7] は 1,750 億パラメータを持つ巨大な汎用言語モデルであり，モデルに与えるプロンプトを適切に設計することで分類問題も解くことができる．本研究では T5 の多言語対応版である mT5 [8] を使用する．

テキスト生成モデルに決まった順番やフォーマットを遵守させるのは難しい．しかし，GitHub Copilot[6] のようなコーディング規約に則ってコードを生成できるシステムの台頭を踏まえると，言語解析への利用も現実的な水準に達しているといえる．

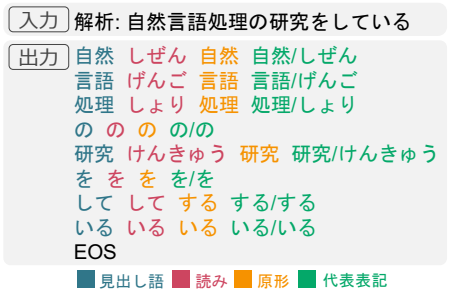

図 3　モデルの入出力フォーマット．EOS は文末を表す特殊記号．

3　提案手法

本節ではまず，テキスト生成モデルで形態素解析を行う手法とモデルの入出力フォーマットについて説明する（3.1 節）．そして提案手法の追加要素である，見出し語強制デコーディング（3.2 節）と疑似データによる曖昧性解消（3.3 節）について説明する．

3.1　テキスト生成モデルによる形態素解析

提案手法では，解析対象の文をテキスト生成モデルに入力し，所定のフォーマットに基づき形態素解析の結果を生成して出力する．モデルの入出力フォーマットを図 3 に示す．入力文の先頭に「解析: 」をつけてからモデルへと入力する．出力フォーマットは Juman++ [5][7] の出力を参考に設計した．各行は見出し語を先頭に読み，原形，代表表記の順に並んでおり，単語ごとに改行することで分かち書きを表現している[8]．

4）　人手によって整備された，単語に対する付加情報．代表表記やカテゴリ，ドメイン等を含む．
5）　非標準表記を正規化する処理（「びみょーー」→「びみょう」）と，連濁により語頭のカナが濁音化した単語を戻す処理（「(上海）ガニ」→「(上海）カニ」）を施した見出し語．
6）　https://github.com/features/copilot
7）　https://github.com/ku-nlp/jumanpp/
8）　分かち書きを生成モデルで行う必要性は大きくないが，分かち書きを行うモジュールを別途使用する際の計算コストを考慮し，本研究では分かち書きも生成モデルで行う．

見出し語は分かち書きされた単語で，読みはその見出し語の読みを平仮名で記したものである．原形は見出し語が活用変化していない場合の形である．代表表記は表記揺れの問題を形態素解析のレベルで吸収するための単語 ID であり，漢字表記やかな表記といった異なる表記でも同一の語であれば同一の代表表記が与えられる[9]．意味情報と代表表記は一対一で対応しているため，代表表記を正しく推定することで意味情報を一意に付与できる．

トークン分類問題で定式化した場合，これらのサブタスクを解くのには煩雑な処理が必要である．提案手法は入出力のフォーマットを定義するだけで，あとは注釈が付与されたコーパスからデータドリブンで学習することができる．

3.2 見出し語強制デコーディング

生成モデルは尤度の高い単語を選んで生成していく．その結果「研究」と入力したにも関わらず，見出し語として「研修」が生成される等，入力文に含まれない文字列を見出し語として生成してしまう場合がある．

そこで見出し語（＝各行の先頭）を生成する際は，入力文に含まれる文字列（サブワード）のみを生成するようにモデルの単語生成確率を補正する．具体的に図 3 の例で説明すると，「自然言語処理の」まで解析し終えたら，入力文のうち未解析の部分文字列（＝「研究をしている」）の先頭と一致するサブワード（「研」「研究」）を語彙の中から探す．この一致したサブワード以外のサブワードの生成確率を 0 にすることで，入力文に含まれる文字列を強制的に見出し語として生成できる．

3.3 疑似データによる曖昧性解消

「子どもとこうえんに行く」という入力中の，かな表記の「こうえん」が「公園」か「講演」かは曖昧性がある．しかしこの「こうえん」の代表表記を「公園/こうえん」と特定することができれば曖昧性を解消でき，意味情報を一意に付与できる．複数サブワードからなる代表表記を分類モデルで推論するのは難しいため， KWJA 1.0 では曖昧性解消は行わず，該当する候補を全て列挙して出力している（上記の例では「公園/こうえん」と「講演/こうえん」）．

本研究では 1 つの見出し語に対し，代表表記を 1 つだけ生成することでこの曖昧性解消を行う．しかしコーパスには曖昧性のある，かな表記の単語が少ないため，かな表記の入力（「こうえん」）から代表表記（「公園/こうえん」）を上手く生成できない．そこで漢字表記の単語をかな表記に置き換えた疑似データを作成し[10]，元データに追加して学習・評価を行う．本研究では漢字表記の名詞を置き換え対象とし，ランダムに選んで置き換える[11]．

表 1 各コーパスの文数

	学習	開発	テスト	合計
KC	13,310	959	1,491	15,760
KWDLC	11,187	1,485	2,006	14,678
Fuman	882	195	202	1,279

表 2 疑似データの統計．各セルの左側が置き換えられた単語を含む文数，右側が置き換えられた単語数である．

	学習	開発	テスト
KC	9,780 / 13,393	708 / 965	1,086 / 1,470
KWDLC	6,002 / 6,757	1,485 / 1,492	1,146 / 1,315
Fuman	326 / 352	75 / 79	76 / 79

4 実験

4.1 コーパスと評価指標

本研究では京都大学テキストコーパス (KC) [9][12]，京都大学ウェブ文書リードコーパス (KWDLC) [10][13]，注釈付き不満買取センターコーパス (Fuman)[14] の 3 種類のコーパスを用いる．モデルへの入力単位は文とし，文の分割境界はコーパスの注釈にならう．表 1 に各コーパスの文数を示す[15]．学習は 3 つのコーパスを全て混ぜて行い，評価は各コーパスごとに行う．評価には単語単位の F1 スコア [5] を用いる．

4.2 モデル

以下の 2 つのモデルを比較する．

Proposed 汎用言語モデルである mT5 [8] の XL モデル[16] を cross entropy loss を損失関数として fine-tuning する[17]．

9） https://usermanual.wiki/Document/manual.643701001.pdf

10） 付録 A に疑似データの作成例を示す．

11） 本研究では置き換える対象を名詞に限定したが，活用のある動詞等への拡張は今後の課題である．

12） https://github.com/ku-nlp/KyotoCorpus

13） https://github.com/ku-nlp/KWDLC

14） https://github.com/ku-nlp/AnnotatedFKCCorpus/

15） 最大系列長は入力が 128 トークン，出力が 512 トークンとし，これらを超える文は学習と評価から除外している．

16） https://huggingface.co/google/mt5-xl

17） 付録 B に実験設定の詳細を示す．

表 3　各コーパスごとの F1 スコア．各セルには「分かち書き / 読み / 原形 / 代表表記」の形式でスコアを記載している．太字は各列内で最もスコアが高いものを示す．上 4 行は疑似データを含まない通常のテストデータに対する結果で，下 2 行は疑似データで置き換えた単語のみに評価対象を限定した結果である．

	KC	KWDLC	Fuman
KWJA	**98.80** / 95.23 / **98.43** / 92.44	**98.40** / 95.65 / 97.90 / 91.51	96.42 / 95.07 / 95.34 / 91.90
Proposed	98.58 / 96.10 / 98.25 / 95.75	97.98 / 96.54 / 97.60 / 94.79	97.31 / 96.64 / 96.57 / 94.70
Proposed + FSD	98.73 / **96.24** / 98.40 / **95.89**	98.31 / **96.88** / **97.93** / **95.07**	**98.15** / **97.45** / **97.42** / **95.24**
Proposed + FSD + DPD	98.36 / 95.77 / 98.00 / 95.26	98.09 / 96.44 / 97.66 / 94.58	97.73 / 96.71 / 96.99 / 95.05
Proposed + FSD	81.72 / 81.72 / 80.32 / 11.26	83.14 / 83.06 / 81.99 / 12.89	94.41 / 94.41 / 93.17 / 19.88
Proposed + FSD + DPD	**93.62** / **93.62** / **93.22** / **72.17**	**94.13** / **94.05** / **93.90** / **72.86**	**96.25** / **96.25** / **96.25** / **78.75**

表 4　出力の入力文復元率（%）．文単位で計算し，入力文が出力の見出し語列と同一であれば「復元」と判定．

	KC	KWDLC	Fuman
Proposed	97.80	97.36	94.03
Proposed + FSD	99.52	99.70	97.01

KWJA　KWJA の Large モデル版を使用する[18]．汎用言語モデルとして RoBERTa [11] Large[19)20)] を採用しており，現時点で公開されている KWJA のモデルの中で最も高精度である．

また Proposed に以下の 2 つの要素を追加した条件でも比較を行う．

Forced Surface Decoding (FSD)　見出し語強制デコーディング（3.2 節）を使用．

Disambiguation using Pseudo Data (DPD)　疑似データ（3.3 節）を元の学習データに加えて学習したモデルを使用．疑似データの統計を表 2 に示す．1 文あたり約 1 個の単語が置き換えられているが，それ以外の単語は元データと同一である．

4.3　結果

表 3 の上 4 行に実験結果を示す．Proposed はコーパスによっては分かち書きと原形で KWJA にやや劣っているものの，読みと代表表記ではどのコーパスでも一貫した精度向上が見られる．提案手法は煩雑な処理を行うことなく，入出力のフォーマットを定義するだけで高い精度を達成できている．

さらに見出し語強制デコーディングを加えた Proposed + FSD では 4 項目全てで精度が底上げされており，分かち書きと原形も KWJA とほぼ同等レベルまで達している．表 4 に出力の入力文復元率を示す．Proposed でも十分高い精度だが，見出し語強制デコーディングを導入することで KC，KWDLC では 99% を上回っており，形態素解析器として実用に耐える復元率になっている．

一方，疑似データを加えた Proposed + FSD + DPD では全体的な精度低下が見られる．疑似データは元データの一部の単語を置き換えることで作成しており，その他の単語は元データと同一である．そうした疑似データを元データに加えて学習した結果，学習される単語に偏りが生じ，全体的に精度が低下したと考えられる．

次に疑似データで置き換えた単語のみに限定して評価した結果を表 3 の下 2 行に示す．疑似データによって全体的に精度が向上しており，特にスコアが低かった代表表記は大きな精度向上が見られる．Proposed + FSD では「こうえん/こうえん」のように，漢字で生成すべき部分を平仮名で生成してしまうケースが多かったが，Proposed + FSD + DPD は「公園/こうえん」のように正しく生成できていた．このように置き換え箇所に対しては一定の精度向上が見られたものの，置き換えていない箇所は精度が低下してしまっている．今後疑似データの作成方法や追加する量等について検討を進める予定である．

5　おわりに

本研究では分類モデルが扱いづらいタスクを対象に，テキスト生成モデルを用いた日本語形態素解析に取り組んだ．実装とタスク設計を単純化したにも関わらず，分かち書きと原形では分類モデルと同水準の精度を出しつつ，読みと代表表記では分類モデルを上回る精度を示した．今後は曖昧性解消に向けた疑似データの活用法について検討を続ける．本研究で提案した手法は KWJA 2.0 として実装・公開予定である[21)]．

18）　KWJA では曖昧性解消を行っていないため代表表記が複数付与されている場合があるが，その場合アルファベット順で最初の代表表記を評価に使用する．

19）　https://huggingface.co/ku-nlp/roberta-large-japanese-char-wwm

20）　https://huggingface.co/nlp-waseda/roberta-large-japanese

21）　付録 C に KWJA 1.0 から 2.0 における主な変更点を示す．

謝辞

本研究で用いた汎用言語モデルの一部は，学際大規模情報基盤共同利用・共同研究拠点 2022 年度公募型共同研究課題「大規模な日本語モデル構築・共有のためのプラットフォームの形成」 (jh221004) において構築した．

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[2] 松田寛. GiNZA – Universal Dependencies による実用的日本語解析. 自然言語処理, Vol. 27, No. 3, pp. 695–701, 2020.

[3] 植田暢大, 大村和正, 児玉貴志, 清丸寛一, 村脇有吾, 河原大輔, 黒橋禎夫. KWJA：汎用言語モデルに基づく日本語解析器. 第 253 回自然言語処理研究会, 京都, 2022.

[4] Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. Improvements of Japanese Morphological Analyzer JUMAN. In **Proceedings of the International Workshop on Sharable Natural Language Resources**, pp. 22–38, 1994.

[5] Arseny Tolmachev, Daisuke Kawahara, and Sadao Kurohashi. Juman++: A Morphological Analysis Toolkit for Scriptio Continua. In **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 54–59, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[6] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. **Journal of Machine Learning Research**, Vol. 21, No. 140, pp. 1–67, 2020.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[8] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In **Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 483–498, Online, June 2021. Association for Computational Linguistics.

[9] Sadao Kurohashi and Makoto Nagao. Building a Japanese Parsed Corpus while Improving the Parsing System. In **Proceedings of the NLPRS**, pp. 719–724, 1998.

[10] 萩行正嗣, 河原大輔, 黒橋禎夫. 多様な文書の書き始めに対する意味関係タグ付きコーパスの構築とその分析. 自然言語処理, Vol. 21, No. 2, pp. 213–247, 2014.

[11] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692, 2019.

[12] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In **International Conference on Learning Representations**, 2019.

[13] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 38–45, Online, October 2020. Association for Computational Linguistics.

[14] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In **International Conference on Learning Representations**, 2021.

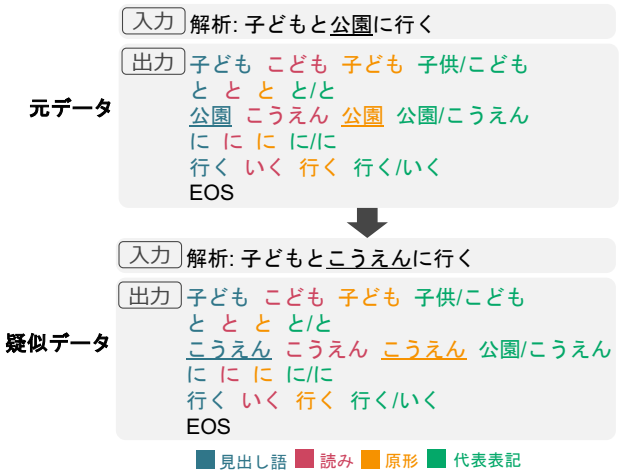

図 4　疑似データの作成例

A　疑似データの例

図 4 に疑似データの作成例を示す．元データの漢字表記の「公園」を，疑似データの入力と出力ではそれぞれかな表記の「こうえん」に置き換えている．

B　実験設定の詳細

バッチサイズは 64 に設定し，最大で 30 エポック学習した．各エポックの終了時に，開発データに対する各コーパスごとの損失を計算し，その平均値が最も低いモデルを採用した．また，3 エポック連続で損失の改善が見られなければ学習を打ち切った．オプティマイザには AdamW [12] を使用し，学習率は 5*e*-4 とした．またウォームアップのステップ数を 250 に設定した，逆平方根学習率スケジューリングを採用した．実装には Hugging Face[22] が提供する transformers [13] 及び Microsoft が提供する DeepSpeed[23] を使用した．学習には NVIDIA A100 80GB 4 枚を使用し約 13 時間かかった．推論時にはビーム幅 3 のビームサーチを行った．

C　KWJA 2.0 に向けて

本論文で提案した手法は統合的日本語解析器 KWJA 2.0 の形態素解析モジュールとして実装する予定である．KWJA 1.0 から 2.0 におけるその他の変更点は以下のとおりである．

学習ベースの文分割　KWJA 1.0 では文分割を規則に基づく文分割器で行っていた．KWJA 2.0 ではこれを学習ベースの文分割器で置き換える．これにより，ウェブ上のテキスト等，くだけた表現に対して頑健な文分割を実現する．

読みアノテーションの修正　JUMAN 系の解析器 [4, 5] では読みの同定は重視されておらず，コーパスのアノテーションにおいても，自動付与された読みの修正は基本的に行わないとされていた．KWJA が文脈に応じた読み推定を行えるようになったことを受けて，読みアノテーションの修正を進めている．

汎用言語モデルにおける DeBERTa の採用　DeBERTa [14] は RoBERTa [11] に対して相対位置埋め込みを含む種々の改善を施したモデルである．KWJA 2.0 では，汎用言語モデルとして RoBERTa の代わりに日本語テキストで事前学習した DeBERTa を採用する．これにより，タスク全体の精度向上が見込まれる．

22）https://huggingface.co/
23）https://github.com/microsoft/DeepSpeed