異なる単語分割システムによる日本語事前学習言語モデルの性能評価

鈴木 雅弘 [1]　坂地 泰紀 [1]　和泉 潔 [1]
[1] 東京大学
msuzuki@g.ecc.u-tokyo.ac.jp　{sakaji,izumi}@sys.t.u-tokyo.ac.jp

概要

日本語で構築された事前学習言語モデルでは，入力文を形態素解析器などを用いて単語に分割してからトークン分割を行うことが多い．しかし，End-to-End で学習を行う近年の事前学習言語モデルにおいて，人為性の高い単語分割を行うことはモデルの効率性を下げる可能性がある．本研究では，異なる単語分割システムから構築した日本語の事前学習言語モデルが，下流の評価タスクの性能に及ぼす影響について検証する．JGLUE ベンチマークによる評価の結果，単語分割システムを用いず構築した言語モデルが，単語分割システムを用いて構築した言語モデルより高い精度を示した．

1　はじめに

近年の自然言語処理では，Transformer [1] をベースとした大規模事前学習言語モデルが盛んに用いられる．BERT [2] を筆頭に，Transformer をベースとした多くのモデルが提案されている [3, 4, 5]．これらのモデルは Wikipedia や Common Crawl など主に英語のコーパスを用いて構築される．言語タスクで高い性能を示すこれらのモデルは，日本語でも構築・公開されるようになってきた．その一方で，英語で提案・構築されたモデルをそのまま日本語に適用する際には，トークン分割での処理を変更することが多い．例えば，BERT や ELECTRA [3] では，入力文を半角スペースで単語に分割し，その後それぞれの単語をトークン (サブワード) に分割する．しかし一般的な日本語の文章では半角スペースで単語が分かれていない．そのため日本語の大規模事前学習言語モデルの構築時には，外部の形態素解析器などを用いて単語分割を行うことが多い．

日本語の大規模事前学習言語モデルの構築例として，東北大学が公開している BERT モデル[1)]では MeCab [6] を，早稲田大学が公開している RoBERTa モデル[2)]では Juman++ [7, 8] を，リクルートが公開している ELECTRA モデル[3)]では Sudachi [9] を用いている．このように，単語分割に用いられるシステムは多岐に渡っている．また，rinna 社の RoBERTa[4)]では，RoBERTa [5] で提案・公開されているモデルと同様に，入力文に単語分割を行わずサブワード分割を行う．日本語で構築されたこれらのモデルを比較する際には，モデルのアーキテクチャだけでなく，用いられている単語分割の手法についても留意する必要がある．この事象は日本語の言語モデル間の比較を困難にしている．

さらに，日本語の言語モデルで用いられている単語分割のシステムは，事前学習言語モデルにおける単語分割が目的ではなく，文の意味処理や情報抽出など，それぞれ目的を異にしている．言語モデルに対して人為性の高い分割を加えることで，モデルの効率が落ちている可能性がある．実際にニューラル機械翻訳では，単語分割をせずに，統計的に一貫性のある分割手法を用いることで精度が向上することが示されている [10]．近年主流の事前学習言語モデルによる言語処理においても，目的に適したトークン分割手法を選ぶ必要があり，従来の単語分割手法に依らずにトークン分割を行うことが下流タスクでの精度の向上につながる可能性がある．

本研究では，日本語において単語分割の手法が事前学習言語モデルに与える影響について検証を行う．具体的には，異なる単語分割の手法を用い，同じコーパスから異なるトークナイザーを構築する．

1) https://huggingface.co/cl-tohoku/bert-base-japanese-v2
2) https://huggingface.co/nlp-waseda/roberta-base-japanese
3) https://huggingface.co/megagonlabs/electra-base-japanese-discriminator
4) https://huggingface.co/rinna/japanese-roberta-base

これらのトークナイザーから構築した事前学習言語モデルについて，下流タスクにおける性能の比較を行う．事前学習言語モデルにおいて，単語分割の手法による下流タスクでの性能やサブワードを含めたトークン分割の傾向の違いが明らかになることで，日本語ドメインにおけるより活発な言語モデルの構築が期待できる．

2 関連研究

本研究と同様に，サブワードを含めたトークン分割システム (トークナイザー) が日本語 BERT モデルに及ぼす影響を評価する取り組みが既になされている [11]．この論文では，IPA 辞書と MeCab を用いて学習した BERT モデルとサブワードの語彙を固定する．サブワードの語彙の再構築や事前学習は行わず，単語分割の手法のみを入れ替え，下流タスクにおける性能の比較を行っている．サブワードの語彙やモデルの重みに変化がない環境では，単語分割手法の違いによって下流タスクでの大きな性能の変化はないことを示している．

汎用言語ドメインとは異なる専門用語が出現する法律や医療，金融のドメインでは，各ドメインの文書を用いて事前学習を行うことでそのドメインのタスクにおいてより高い性能を示す [12, 13, 14]．その際，事前学習のコーパスだけでなくトークナイザーを構築するコーパスもドメイン適合を行うことで，より自然なトークン分割が可能になりモデルの性能の向上にも寄与する [14] など，語彙のドメイン適合も性能に影響を与える [15] ことが示されている．このように，モデルのアーキテクチャだけでなくトークナイザーの構築方法も重要であることから，トークナイザーの構成要素の 1 つである単語分割も重要な要素と捉えることができる．

3 トークナイザーの構築

近年の事前学習言語モデルのトークン分割 (トークナイザーの適用) は次の手順で行う．まず単語分割を行う場合には入力文を単語に分割する．その後，それぞれの単語をトークン (サブワード) に分割を行う．本節では，実験で適用するサブワード分割と単語分割の手法についてそれぞれ述べる．

3.1 サブワード分割

Transformer をベースとした事前学習言語モデルのトークン分割の手法としては，WordPiece [16] と SentencePiece [10] の 2 種類に大きく分けることができる．本研究ではサブワード分割の実装に，実装が公開されている SentencePiece を用いる．SentencePiece ではスペースも他の文字と同様に扱い，スペースを含めた一文に対してサブワードに分割する．WordPiece と異なり，単語分割の前処理が必要ないため，日本語でも単語分割器を用いずにトークナイザーを構築できる．本研究では，単語分割の手法の比較に加え，単語分割をせずに入力文に直接 SentencePiece を適用する場合についても比較を行う．

表 1 比較を行う単語分割器．モデル名は，単語分割器ごとに構築した言語モデルの呼称に用いる．Sudachi の辞書は Core を用いる．Unsegmented は，単語分割をせずに入力文に直接 SentencePiece を適用することを指す．

モデル名	システム	辞書	分割モード
MeCab-IPA	MeCab	IPA 辞書	-
MeCab-Unidic	MeCab	Unidic	-
Juman++	Juman++	-	-
Sudachi-A	Sudachi	-	A
Sudachi-B	Sudachi	-	B
Sudachi-C	Sudachi	-	C
GSDLUW	GSD LUW	-	-
Unsegmented	なし	-	-

3.2 単語分割

単語分割を行うために，本研究では以下の 4 種類の手法を用いる．

- MeCab
- JUMAN++
- Sudachi
- UD Japanese GSD LUW[5] [17]

これらはそれぞれ本来の役割や目的が異なるものの，本研究では以降「単語分割器」として統一した呼称にて扱うこととする．UD Japanese GSD LUW は，Universal Dependencies (UD) Japanese に基づいて国語研長単位 (Long Unit Word, LUW) [18] の解析を行うモデルである．UD Japanese GSD LUW の当該バージョンはプレリリースであることに留意されたい．

単語分割器によっては手法や辞書ごとに異なる単語分割が行われる場合があるため，これについても比較を行う．MeCab では，システム辞書として IPA

5) https://github.com/megagonlabs/UD_Japanese-GSD/releases/tag/r2.9-NE

表 2 構築したトークナイザーを用いたトークン分割の例．トークン分割の際，サブワード分割のために単語間に挿入されたメタ文字"▁" (U+2581) は除去し，当該メタ文字のみで構成されたトークンも除去している．

モデル名	単語分割	トークン分割
MeCab-IPA	ラテンアメリカ/諸国/から/の/外交/官/や/研修生/を/養成/し/て/いる/。	ラテンアメリカ/諸国/から/の/外交/官/や/研修/生/を/養成/し/て/いる/。
MeCab-Unidic	ラテンアメリカ/諸国/から/の/外交/官/や/研修/生/を/養成/し/て/いる/。	ラテンアメリカ/諸/国/から/の/外交/官/や/研/修/生/を/養/成/し/て/いる/。
Juman++	ラテン/アメリカ/諸国/から/の/外交/官/や/研修/生/を/養成/して/いる/。	ラテン/アメリカ/諸国/から/の/外交/官/や/研修/生/を/養成/して/いる/。
Sudachi-A	ラテン/アメリカ/諸国/から/の/外交/官/や/研修/生/を/養成/し/て/いる/。	ラテン/アメリカ/諸国/から/の/外交/官/や/研修/生/を/養/成/し/て/いる/。
Sudachi-B	ラテン/アメリカ/諸国/から/の/外交官/や/研修生/を/養成/し/て/いる/。	ラテン/アメリカ/諸国/から/の/外交官/や/研修/生/を/養成/し/て/いる/。
Sudachi-C	ラテンアメリカ/諸国/から/の/外交官/や/研修生/を/養成/し/て/いる/。	ラテンアメリカ/諸国/から/の/外交官/や/研修/生/を/養成/し/て/いる/。
GSDLUW	ラテンアメリカ諸国/から/の/外交官/や/研修生/を/養成し/ている/。	ラテンアメリカ/諸国/から/の/外交官/や/研修/生/を/養成/し/ている/。
Unsegmented	-	ラテンアメリカ/諸国/からの/外交官/や/研修/生/を/養成/している/。

辞書と UniDic が利用可能である．また，Sudachi では 3 つの異なる分割モード (A:UniDic 短単位相当，C:固有表現相当，B:A と C の中間的な単位) を利用できる[6]．本研究では辞書や分割モードが異なるこれらの場合についても比較を行う．本研究で構築するトークナイザーの一覧を表 1 に示す．上述した単語分割器のバリエーションに加え，3.1 節で述べた，単語分割をせず入力文に SentencePiece をそのまま適用するものについても表 1 の最下部に Unsegmented として記載している．

3.3 トークナイザーの振る舞いの比較

構築したトークナイザーを用い，同じ文について単語分割を行った例を表 2 に示す．この例では，単語分割を行わない Unsegmented でも他の単語分割器と比べて大きく異なることなくトークン分割が行われている．1 文から生成されたトークン数は 11 と最も少なく，次に少なかったのは GSDLUW の 13 トークンであった．入力文に直接 SentencePiece を適用することで，単語分割器を適用した場合に比べてより少ないトークン数で分割が行われていると考えられる．トークン数が多かったのは MeCab-Unidic (18 トークン) と Sudachi-A (17 トークン) であった．これらの単語分割器による分割では，MeCab-Unidic で 15 単語，Sudachi-A で 16 単語と単語分割の時点から語数が多い．これはどちらも短単位 [19] にもとづいているために比較的短い語ごとに分割される事によると考えられる．

4 事前学習言語モデルの構築

事前学習モデルのアーキテクチャとして，本研究では RoBERTa [5] を用いる．学習率は 1e-4，バッチサイズは 4,024，学習ステップ数は 12,000，Warm up ステップ数は 640 とする．学習速度向上のため，計算精度は bfloat16 とし，実装には DeepSpeed [20] を活用する．その他のハイパーパラメータは文献 [5] に記載されている base モデルを参考とする．

学習を行うコーパスには日本語 Wikipedia [7] の 2022 年 6 月 1 日の版から抽出した約 3,400 万文を用いる．モデルの最長入力長は 128 とする．文献 [5] と異なり，データセットには予めマスキングを行う．

5 評価実験

前節で構築した事前学習 RoBERTa モデルに対し，ファインチューニングによる評価実験を行い性能を評価する．本研究では，JGLUE [21, 22] に含まれる JSTS，JNLI，JCommonsenseQA を用いて評価を行う．JGLUE で公開されているデータとしては，これらの他に MARC-ja と JSQuAD が存在するが，これらのタスクの入力長の最大値が 128 を超えていることから，上記の 3 タスクに限定する．2023 年 1 月現在，JGLUE では学習データと評価データのみが公開されており，テストデータが公開されていない．そのため，JGLUE における評価データを本研究ではテストデータとし，JGLUE における学習データを本研

6) https://github.com/WorksApplications/Sudachi/blob/v0.7.0/README.md#分割モード

7) https://dumps.wikimedia.org/jawiki/

表 3　評価実験のハイパーパラメータ

ハイパーパラメータ	値
学習率	{5e-6, 1e-5, 2e-5, 3e-5}
最大エポック数	10
バッチサイズ	{16, 32}
最大入力長	128 (JSTS, JNLI), 64 (JCommonsenseQA)
Warmup Ratio	0.1

究では学習データと評価データに分ける．本研究で使用する評価データのサンプル数は，テストデータ (JGLUE における評価データ) と同数[8]とする．

評価実験で使用するハイパーパラメータを表 3 に示す．各エポックごとに評価を行い，最も精度が高かったエポックを採用する．構築したそれぞれのモデルについて，異なる 10 個のシード値を用いて実験を行い，算出された 10 個の結果について平均したものを報告する．

6　結果と考察

評価実験の結果を表 4 に示す．評価実験を行った 3 つのタスクの全てで評価指標は同じではないものの，比較のために 3 つのタスクの結果の平均をとり，右列に記載している．3 つのタスクの平均精度が最も高かったのは，入力文に直接 SentencePiece を適用した Unsegmented のモデルとなった．Unsegmented モデルでは JSTS や JNLI は他のモデルと精度は大きく変わらなかったものの，JCommonsenseQA (JCQA) において他のモデルより高い精度を示した．JCommonsenseQA で精度の差が見られた理由としては，学習データの数とタスクの難易度が考えられる．JCommonsenseQA は JSTS と JNLI に比べ学習データ数が少なく，また常識 (Common Sense) を問う問題であり，学習データのファインチューニングのみによって適合することは他の 2 つのタスクに比べ困難である．そのため，事前学習を効率的に行うことができたモデルがより高い精度を示した可能性がある．

単語分割を行ったモデルでは，3 つのタスクごとに性能差は存在したものの，平均して大きく性能差が見られたモデルはなかった．大きな性能差が見られなかった理由として，本研究で扱った下流タスクが分類や類似度判定タスクのみであったことが考えられる．単語の認識がより重要になると考えられる固有表現抽出や構文解析のタスクでは，単語分割器によって異なる傾向が見られる可能性がある．

8）各サンプル数は https://github.com/yahoojapan/JGLUE/tree/v1.1.0#tasksdatasets を参照されたい

表 4　評価実験の結果．JCQA は JCommonsenseQA を表す．評価指標は，JSTS では Pearson，JNLI・JCQA では Accuracy を用いる．

モデル名	JSTS	JNLI	JCQA	平均
MeCab-IPA	.851	.793	.625	.757
MeCab-Unidic	.849	.796	.615	.753
Juman++	**.855**	.802	.605	.754
Sudachi-A	.849	**.805**	.619	.758
Sudachi-B	.850	.804	.620	.758
Sudachi-C	.851	.800	.612	.754
GSDLUW	.848	.802	.621	.757
Unsegmented	.852	.804	**.646**	**.768**

本研究ではサブワード分割手法として SentencePiece のみを用いたが，BERT や ELECTRA では WordPiece が用いられている．異なるサブワード分割手法を用いた際の下流タスクでの性能についての検証は今後の課題である．また，本研究では入力長の制限から評価を行うタスクが 3 種類であった．より精緻な比較を行うためには，評価タスクの種類を増やすことも重要であると考えられる．

7　おわりに

本研究では，日本語の大規模事前学習言語モデルにおいて様々な単語分割器が用いられていることを念頭に，単語分割器が下流タスクの性能に与える影響の比較を行った．単語分割器ごとに異なるトークン分割の挙動を示すトークナイザーを構築した．これらのトークナイザーを用いて事前学習言語モデルを構築し，JGLUE ベンチマークを用いた評価実験を行った．その結果，単語分割器を用いず SentencePiece のみでトークナイザーを構築した言語モデルが他の単語分割器を用いた言語モデルに比べて高い精度を示した．単語分割器を用いた言語モデルの間で大きな精度の差は見られなかった．

単語分割器を用いずにトークン分割を行うことで，人間の直感とは異なる分割が行われる可能性がある．しかし，AlphaGo Zero [23] が人間を越え新しい定石を生み出してきたように，言語処理においても今までの単語分割を用いない分割手法が性能向上をもたらすことが示唆された．本研究が日本語における最先端の事前学習言語モデルのさらなる活用につながることを期待する．

謝辞

本研究はJSPS科研費JP21K12010，JST未来社会創造事業JPMJMI20B1，及びJSTさきがけJPMJPR2267の助成を受けたものです．

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In **Advances in Neural Information Processing Systems (NeurIPS)**, Vol. 30, pp. 5999–6009, 2017.

[2] 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. Association for Computational Linguistics, 2019.

[3] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In **8th International Conference on Learning Representations (ICLR)**, 2020.

[4] Radford, Alec and Narasimhan, Karthik and Salimans, Tim and Sutskever, Ilya. Improving Language Understanding by Generative Pre-Training, 2018.

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

[6] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying Conditional Random Fields to Japanese Morphological Analysis. In **Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)**, pp. 230—237, 2004.

[7] Hajime Morita, Daisuke Kawahara, and Sadao Kurohashi. Morphological Analysis for Unsegmented Languages using Recurrent Neural Network Language Model. In **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**, pp. 2292–2297, 2015.

[8] 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, 2018.

[9] Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. Sudachi: a Japanese Tokenizer for Business. In **Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)**, 2018.

[10] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 66–71, 2018.

[11] 築地俊平, 新納浩幸. Tokenizer の違いによる日本語 BERT モデルの性能評価. 言語処理学会第 27 回年次大会, 2021.

[12] Keisuke Miyazaki, Hiroaki Yamada, and Takenobu Tokunaga. Cross-domain analysis on Japanese legal pretrained language models. In **Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022**, pp. 274–281, 2022.

[13] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. In **Conference on Health, Inference, and Learning (CHIL) Workshop Track**, 2020.

[14] Masahiro Suzuki, Hiroki Sakaji, Masanori Hirano, and Kiyoshi Izumi. Constructing and analyzing domain-specific language model for financial text mining. **Information Processing & Management**, Vol. 60, No. 2, p. 103194, 2023.

[15] Shoetsu Sato, Jin Sakuma, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. Vocabulary Adaptation for Domain Adaptation in Neural Machine Translation. In **Findings of the Association for Computational Linguistics: EMNLP 2020**, pp. 4269–4279, November 2020.

[16] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, 2016.

[17] 松田寛, 大村舞, 浅原正幸. Ud japanese に基づく国語研長単位解析系の構築. 言語処理学会第 28 回年次大会, 2022.

[18] 小椋秀樹, 小磯花絵, 冨士池優美, 宮内佐夜香, 小西光, 原裕. 『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（上）, 2011.

[19] 小椋秀樹, 小磯花絵, 冨士池優美, 宮内佐夜香, 小西光, 原裕. 『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（下）, 2011.

[20] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. DeepSpeed: System Optimizations Enable Training Deep Learning Models with Over 100 Billion Parameters. In **Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**, KDD '20, pp. 3505–3506, 2020.

[21] Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. JGLUE: Japanese general language understanding evaluation. In **Proceedings of the Thirteenth Language Resources and Evaluation Conference**, pp. 2957–2966, 2022.

[22] 栗原健太郎, 河原大輔, 柴田知秀. JGLUE: 日本語言語理解ベンチマーク. 言語処理学会第 28 回年次大会, 2022.

[23] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. **Nature**, Vol. 550, No. 7676, pp. 354–359, 2017.